N_COM_DAD_A3 - Texto de apoio

Site: EAD Mackenzie
Tema: COMUNICAÇÃO DE DADOS {TURMA 03B} 2023/1
Livro: N_COM_DAD_A3 - Texto de apoio

Impresso por: FELIPE BALDIM GUERRA .
Data: domingo, 9 abr 2023, 05:04

Índice

1. CAMADA DE APLICAÇÃO

Quando pensamos na Internet, geralmente, vem em nossa cabeça uma série de serviços criados ao longo dos anos que facilitaram e transformaram nossas vidas.

Atualmente, comprar um produto ou uma refeição pela Internet e esperar por ela no conforto da sua casa ou escritório, assim como chamar um serviço de viagens de carro em detrimento a procurar um taxi ou, até mesmo, passar horas assistindo a filmes ou a conteúdos sob demanda são práticas cada vez mais comuns em nossa sociedade. Todos esses serviços são aplicações que usam a infraestrutura de redes para aproximar as pessoas e facilitar nossas vidas. Segundo a visão do Forouzan (2006), a Internet inteira, tanto hardware como software, foi projetada e desenvolvida para fornecer serviços aos usuários na camada de aplicação.

Para iniciar nossos estudos sobre o modelo TCP/IP, vamos nos debruçar sobre a camada de aplicação, na qual nosso ponto de partida apresentará uma visão sobre os dilemas e paradigmas dessa camada e nosso texto de apoio apresentará o protocolo HTTP, os serviços de correio eletrônico, bem como o sistema de resolução de domínio.

1.1. World Wide Web e o protocolo HTTP

O WWW (World Wide Web) é um dos serviços mais utilizado na Internet (Comer, 2016). Conhecida como Web, o WWW é uma estrutura arquitetônica que permite o acesso a documentos vinculados espalhados por milhões de máquinas na Internet (Kurose, 2013).

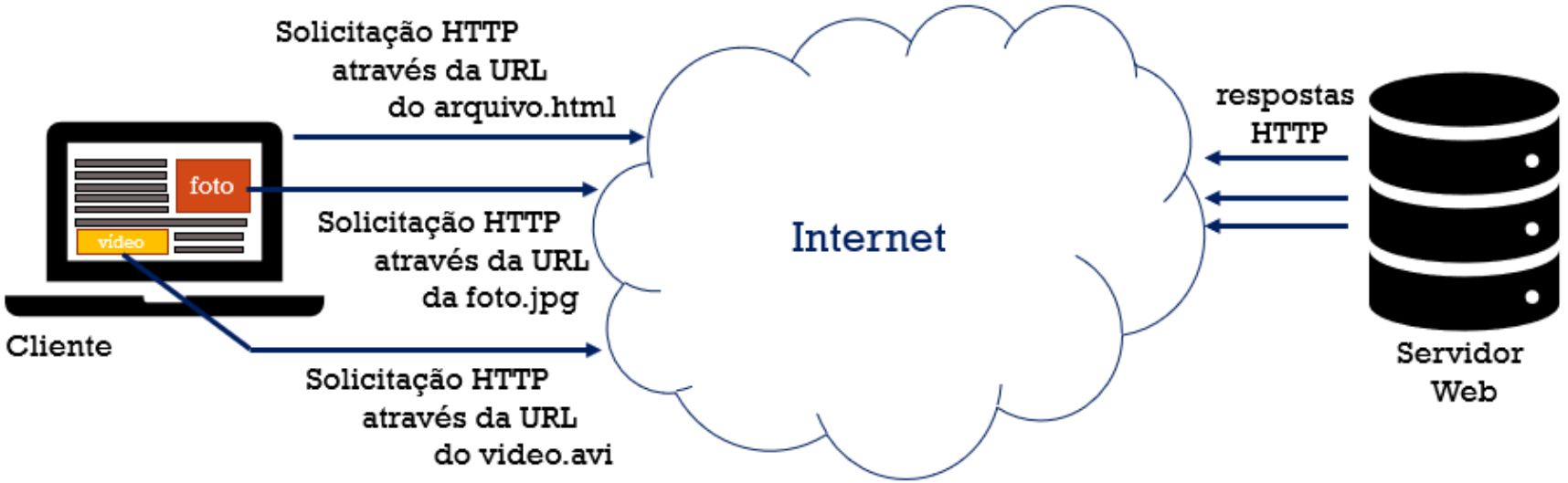
Criada no início dos anos 1990 pelo pesquisador britânico Tim Berners-Lee (Timothy John Berners-Lee), a Web tinha como objetivo criar uma rede de documentos integrados (relatórios, fotos e outros documentos relevantes) por meio de links (Tanenbaum, 2010) que permitisse a integração e contribuição de pesquisadores de diferentes países de maneira mais eficiente. Além de integrar informações por links, a web permitiria a oferta de conteúdo sob demanda.

A oferta sob demanda talvez tenha sido o elemento motivador que levou Marc Andreessen a desenvolver o primeiro navegador Web (Mosaic, lançado em 1993) e, posteriormente, fundar a Netscape Communications Corp. Certamente, a fundação da Netscape foi um marco importante da Internet, pois possibilitou aos usuários fora do mundo acadêmico o acesso à Web e ao conteúdo de seus documentos conhecidos como páginas Web.

As páginas Web são constituídas de um arquivo HTML (HyperText Markup Language) que pode incorporar textos e hipertextos, bem como diversos outros objetos conhecidos como hipermídias (fotos, sons, vídeos, applet java etc.) que podem ser acessados por uma URL (Uniform Resource Locator).

O acesso às páginas Web por URLs é realizado por meio de uma estrutura cliente-servidor. O formato e a sintaxe das mensagens, assim como as regras de interação entre cliente e servidor, são definidos pelo protocolo HTTP (HyperText Transfer Protocol) . A figura 1 ilustra a arquitetura da Web.

Figura 1 – Arquitetura Web



Na figura 1, podemos ver uma representação de um cliente usando o protocolo HTTP para realizar solicitações de uma página web, bem como realizando as solicitações dos objetos da página através de suas respectivas URLs.

Para ocorrer essa interação entre cliente e servidor, as aplicações que irão se comunicar pelo protocolo HTTP são:

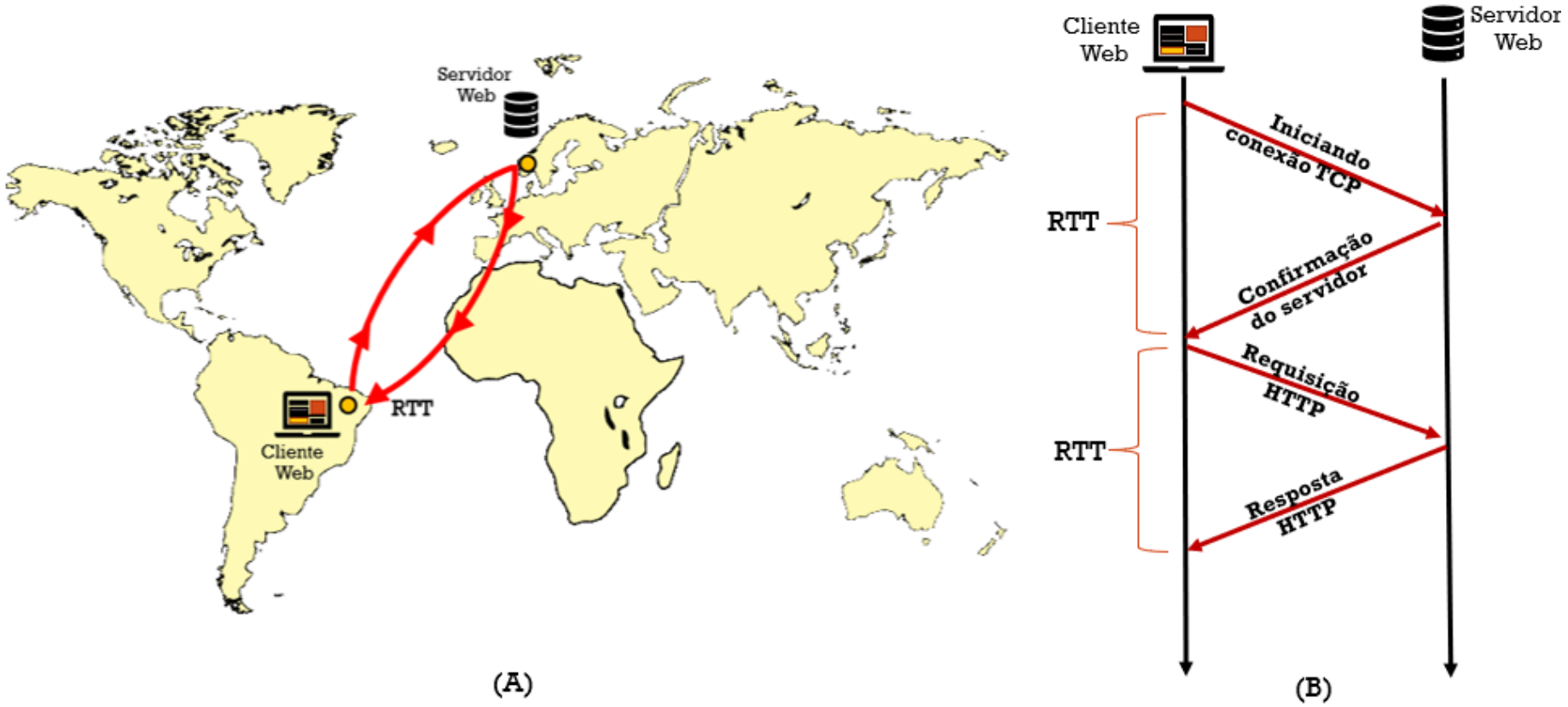
- Agente do usuário – conhecido popularmente como web browser, é o responsável por realizar as solicitações dos objetos de uma página web.
- Servidor Web – é o responsável pode receber as solicitações e responder ao cliente com o objeto solicitado ou com uma mensagem de erro.

- Conexões persistentes e não persistentes

O protocolo HTTP usa como protocolo de transporte o TCP (Transmission Control Protocol) que possui, como uma de suas características, a necessidade de estabelecer conexão entre cliente e servidor antes de qualquer ação dos protocolos da camada de aplicação. Para estabelecer conexão, o cliente sinaliza pelo protocolo TCP que irá se conectar ao servidor para demandar um serviço específico. O servidor, por sua vez, responde a essa solicitação indicando se o serviço solicitado está ou não disponível.

O tempo que leva para estabelecer conexão ou o tempo que leva para uma informação qualquer chegar a um servidor e retornar ao cliente é chamado de Round trip time (RTT) (Figura 2A). O RTT pode variar em função da largura de banda, do atraso de propagação (tempo necessário para um pacote viajar de um ponto de entrada a um ponto de saída) e dos atrasos decorrentes do processamento dos pacotes em cada nó do núcleo da rede. Pensando em nossas solicitações HTTP, como podemos observar na Figura 2B, para solicitar um objeto Web, serão necessários dois RTTs: um RTT para estabelecer conexão entre cliente e servidor e outro RTT para fazer a requisição do objeto.

Figura 2 – RTT: Tempo necessário para um pacote sair de um cliente até o servidor e retornar (A) e representação de solicitação Web usando dois RTTs



Na primeira versão do protocolo HTTP (HTTP 1.0), usada principalmente quando as páginas Web eram simplesmente um arquivo HTML com textos e hiperlinks, o TCP estabelecia uma conexão com o servidor, o protocolo HTTP realizava a solicitação da página web e, após receber o objeto solicitado, a conexão entre cliente e servidor era encerrada. Com a evolução das páginas Web, dezenas de objetivos e links foram adicionados a essas páginas e o uso de conexões não persistentes se tornou muito dispendiosa, pois, para requisitar cada objeto da página Web, era necessário estabelecer uma conexão diferente com o servidor que seria finalizada após o recebimento desse único objeto.

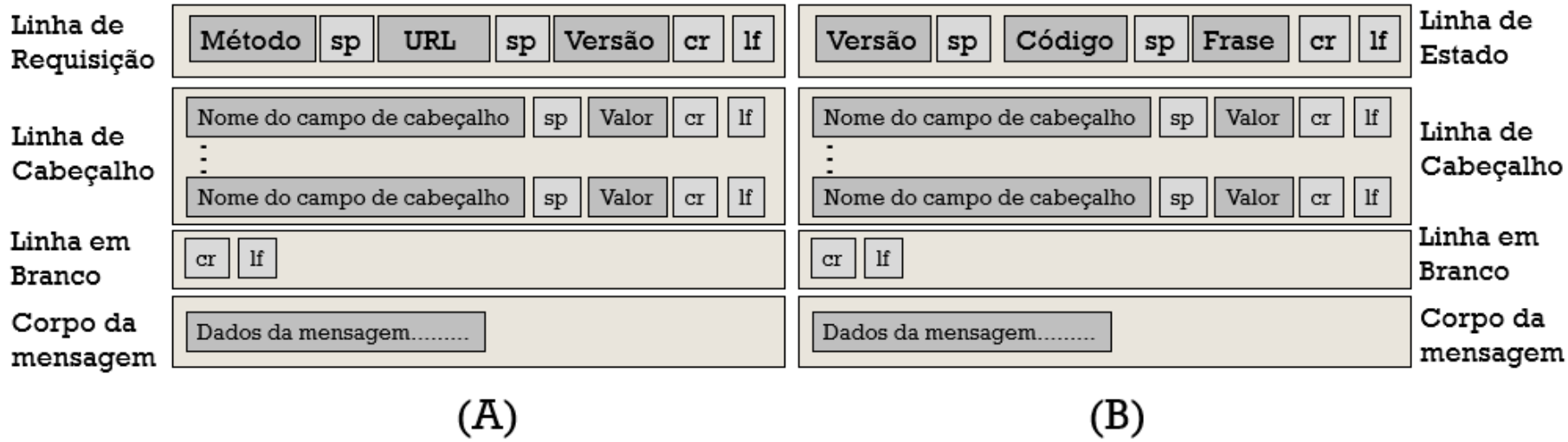
Na versão HTTP 1.1, os cientistas da computação reduziram o tempo total despendido para recebimento da página Web, diminuindo o número de RTTs por meio do uso de conexões persistentes. Nessa nova versão, o protocolo HTTP pode realizar mais de uma solicitação ao servidor Web usando a mesma conexão, o que reduz o tempo gasto para iniciar novas conexões, bem como reduz a carga de processamento (overhead) sobre

o sistema operacional (SO) responsável por gerenciar as conexões TCP.

- Formato das mensagens

Como já citado, o protocolo HTTP define o formato tanto das mensagens de requisição de uma página Web quanto das respostas enviadas pelo servidor para essas solicitações. Na Figura 3, podemos observar o formato da mensagem de requisição do HTTP (Figura 3A), assim como o formato da mensagem de resposta (Figura 3B).

Figura 3 – Formato da mensagem de requisição (A) e resposta (B) do protocolo HTTP



A primeira linha da requisição HTTP é composta por um campo método que define o tipo de solicitação que será realizada, um campo com a URL do objeto que será solicitado e um campo com a versão do HTTP utilizada pelo agente do usuário. A linha de requisição, assim como as demais linhas, é finalizada por dois caracteres: retorno de carro (carriage return) e nova linha (line feed).

Como exemplos de possibilidades usadas no campo método, temos o GET, responsável por solicitar um objeto específico ao servidor, o método POST, responsável por enviar informações para o servidor (por exemplo, envio de formulários pela Web) e o método DELETE que permite ao cliente remover um objeto no servidor se tiver permissão para fazê-lo. Os principais métodos utilizando pelo HTTP 1.1 são apresentados

na Tabela 1.

Método	Descrição
GET	Solicita recurso específico.
HEAD	Solicita informações específicas do recurso.
POST	Submete algum recurso ao servidor.
PUT	Submete algum recurso ao servidor.
DELETE	Remove objetos do servidor.
TRACE	Depura dados.
CONNECT	Conecta a um servidor web por meio de intermediário (*proxy*).
OPTIONS	Solicita propriedades da página.

As linhas de cabeçalhos da requisição HTTP consistem em informações sobre o cliente que serão interpretadas pelo servidor. Podemos citar, como exemplos de informações, a identificação do agente do usuário (User-agent), o formato de mídias suportados pelo cliente (Accept), os idiomas que o cliente pode aceitar (Accept-language), as codificações que o cliente pode manipular (Accept-encoding) ou as informações usadas pelo caching (If-Modified-Since).

Finalizando nossa análise ao cabeçalho da mensagem de requisição do HTTP, temos o corpo da mensagem, onde eventuais dados podem ser anexados para serem enviados ao servidor.

A mensagem de resposta do servidor a uma requisição HTTP (Figura 3B) começa pela linha de estado, em que o primeiro campo define a versão do protocolo HTTP utilizado, e o campo código de status define o estado do pedido. A Tabela 2 apresenta os grupos de código de estados bem como exemplos de códigos que podem ser enviados pelo servidor em respostas a uma solicitação web.

Código	Significado	Exemplo
1xx	Informação	100 = Servidor concorda em tratar solicitação
2xx	Sucesso	200 = Solicitação com sucesso 204 = Nenhum conteúdo presente
3xx	Redirecionamento	301 = Objeto movido 304 = Conteúdo não modificado (Webcache)
4xx	Erro no cliente	400 = Erro na requisição (bad request) 401 = Não autorizado 404 = Objeto não localizado
5xx	Erro no servidor	500 = Erro interno do servidor 503 = Tente novamente mais tarde

Fonte: Tanembaum (2010).

As linhas de cabeçalhos da mensagem de resposta HTTP contêm informações sobre o servidor que serão enviadas para o cliente, como qual aplicação está em execução no servidor (Server), a data atual (Date) e a data da última alteração do objeto enviado (Last-modified) ou o esquema de codificação da mensagem enviada (Content-Encoding).

As informações mais detalhadas sobre os métodos e cabeçalhos da mensagem de requisição, assim como os significados dos códigos e linhas de cabeçalho da resposta do protocolo HTTP podem ser encontradas no site da Organização Força-Tarefa de Engenharia da Internet (IETF – Internet Engineering Task Force), assim como nas bibliografias do curso como Forouzan (2006) e Tanenbaum (2010).

- **Cookies**

O protocolo HTTP é um protocolo sem estado, ou seja, quando um cliente faz uma solicitação ao servidor Web, o servidor responderá a essa solicitação e não guardará informações sobre o cliente. Uma maneira de criar estado ao protocolo HTTP é o uso de cookies que têm como objetivo identificar e monitorar as atividades de seus usuários (Kurose, 2013).

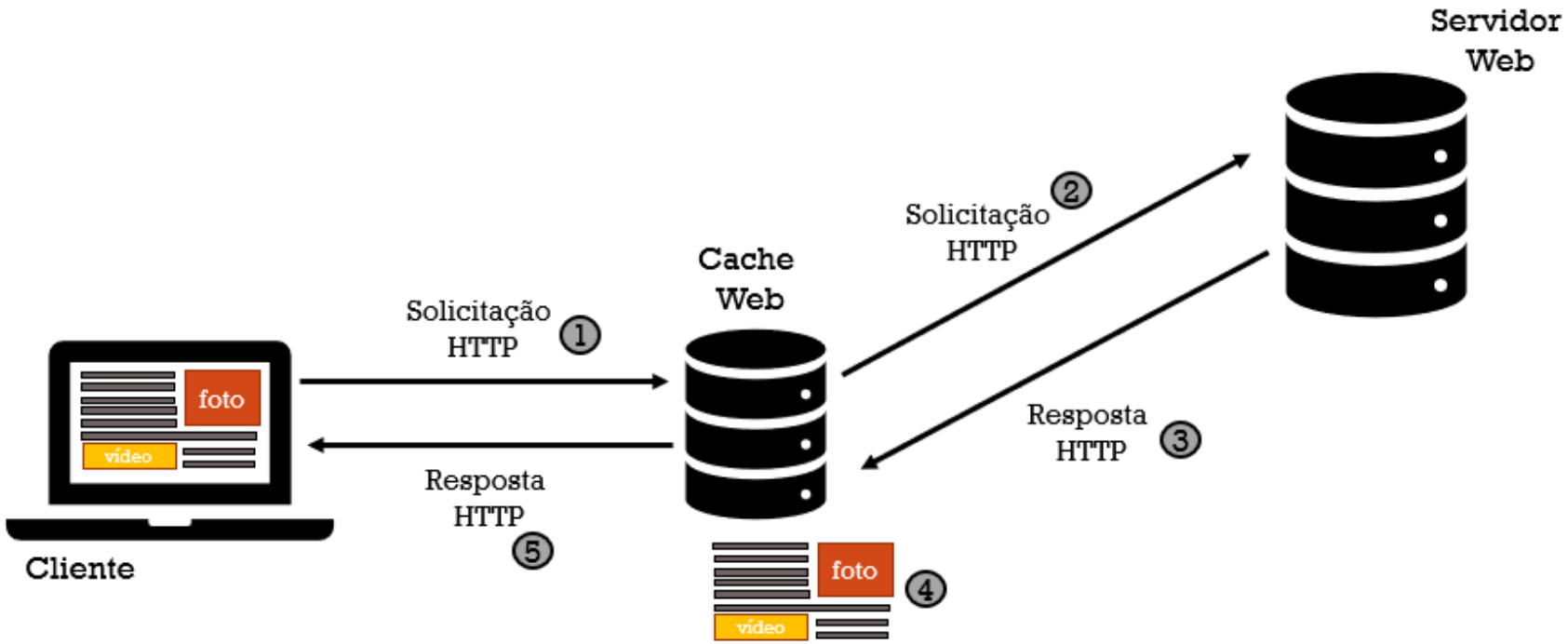
Os cookies podem ser utilizados em diversos serviços, dentre os quais podemos citar os sites de e-commerce, onde os cookies permitem que um usuário possa adicionar produtos no carrinho de compras virtuais e continue navegando no site sem perder as informações do carrinho, ou aplicações redes sociais que identificam seus usuários mesmo sem a necessidade de autenticar (digitar seu usuário e senha) todas as vezes que acessarem o site da rede social.

Os cookies que armazenam informações sobre o histórico de navegações, palavras chaves utilizadas em pesquisas realizadas em buscadores web ou, até mesmo, interesses explícitos em redes sociais podem ser usados por empresas de marketing para traçar um perfil do usuário e, com isso, alimentar banners de publicidades disponíveis nos sites que esse usuário visitar.

- **Caches Web (Servidor Proxy)**

Caches Web ou servidores proxy são um servidor que tem como objetivo intermediar uma solicitação entre um cliente e o servidor Web. Dotado de uma unidade de armazenamento, além de intermediar as solicitações Web do cliente, o cache Web armazena e mantém uma cópia dos objetos requisitados recentemente (Kurose, 2013) em sua unidade de armazenamento conforme representado na Figura 4.

Figura 4 – Representação de requisição HTTP com cache Web



Na Figura 4, vemos que o cliente HTTP, ao invés de solicitar uma página Web diretamente ao Servidor Web, realiza a solicitação para o cache Web (1), que receberá a solicitação e gerará uma nova solicitação para o servidor Web (2). O Servidor Web responderá à solicitação do cache Web (3) que, por sua vez, repassará a resposta ao cliente (5), salvando uma cópia dos objetos recebidos (4) em sua unidade de armazenamento. Caso outro cliente requisite uma página web já armazenada no cache em solicitações anteriores, o cache web entrega a cópia da página web que está em sua unidade de armazenamento, reduzindo, assim, a latência e o tráfego no enlace de saída da rede.

Para manter o cache web sempre atualizado, o protocolo HTTP usa o recurso chamado GET condicional. No GET condicional, mesmo quando o servidor intermediário possui em sua unidade de armazenamento a página solicitada por um cliente, o cache web enviará uma requisição GET para o servidor web, adicionando no cabeçalho da solicitação uma linha com a informação “If-modified-since” e a data da versão da última atualização do arquivo que ele tem salvo. Ao receber uma solicitação contendo a linha “If-modified-since”, o servidor web comparará a data

recebida com a data do arquivo que ele possui. Caso o arquivo não tenha sofrido nenhuma atualização, o servidor web retorna ao cache web uma mensagem com código de status 304 (Not Modified), se não, o servidor web retorna uma mensagem com código de status 200 e, no corpo da mensagem, o objeto atualizado.

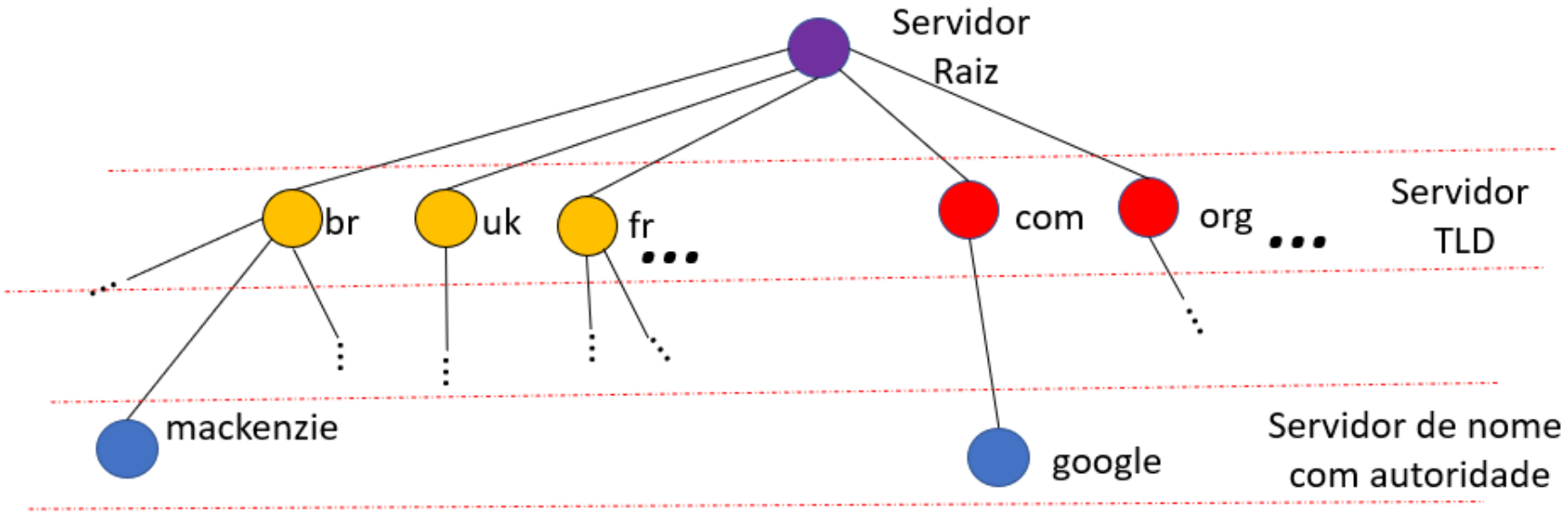
1.2. Sistema de nomes de Domínio

Diariamente, acessamos dezenas de sites e serviços na Internet. Na Aula 2, vimos que o protocolo IP é o componente que permite endereçar os dispositivos, assim como definir como sistemas finais e roteadores determinarão o destino das mensagens com base no cabeçalho da camada de rede. Pensando nessa característica de nossa arquitetura de redes, imagine você ter de saber o endereço IP de todo servidor, host ou serviço que acessa em seu dia a dia.

Para evitar que o usuário precise decorar todos os endereços IP dos servidores que acessa, foi criado, em 1983, o sistema de nomes de domínios – DNS (Domain Name System) (Tanenbaum, 2010). O DNS é um serviço de rede que tem como objetivo mapear nomes de domínios para seus respectivos endereços IP por meio de uma estrutura hierárquica de banco de dados distribuído, ou seja, quando você digitar em seu navegador web uma URL, como “www.mackenzie.br”, o navegador executará uma aplicação chamada resolvedor (Tanenbaum, 2010) que consultará a base de dados do DNS e receberá como retorno o endereço IP associado à URL consultada.

A base de dados distribuída usada pelo DNS consiste em diversos servidores espalhados pelo mundo, organizados em uma topologia do tipo árvore, na qual, em uma abordagem mais simples, a hierarquia desses servidores pode ser dívida em três classes, conforme representado na Figura 5.

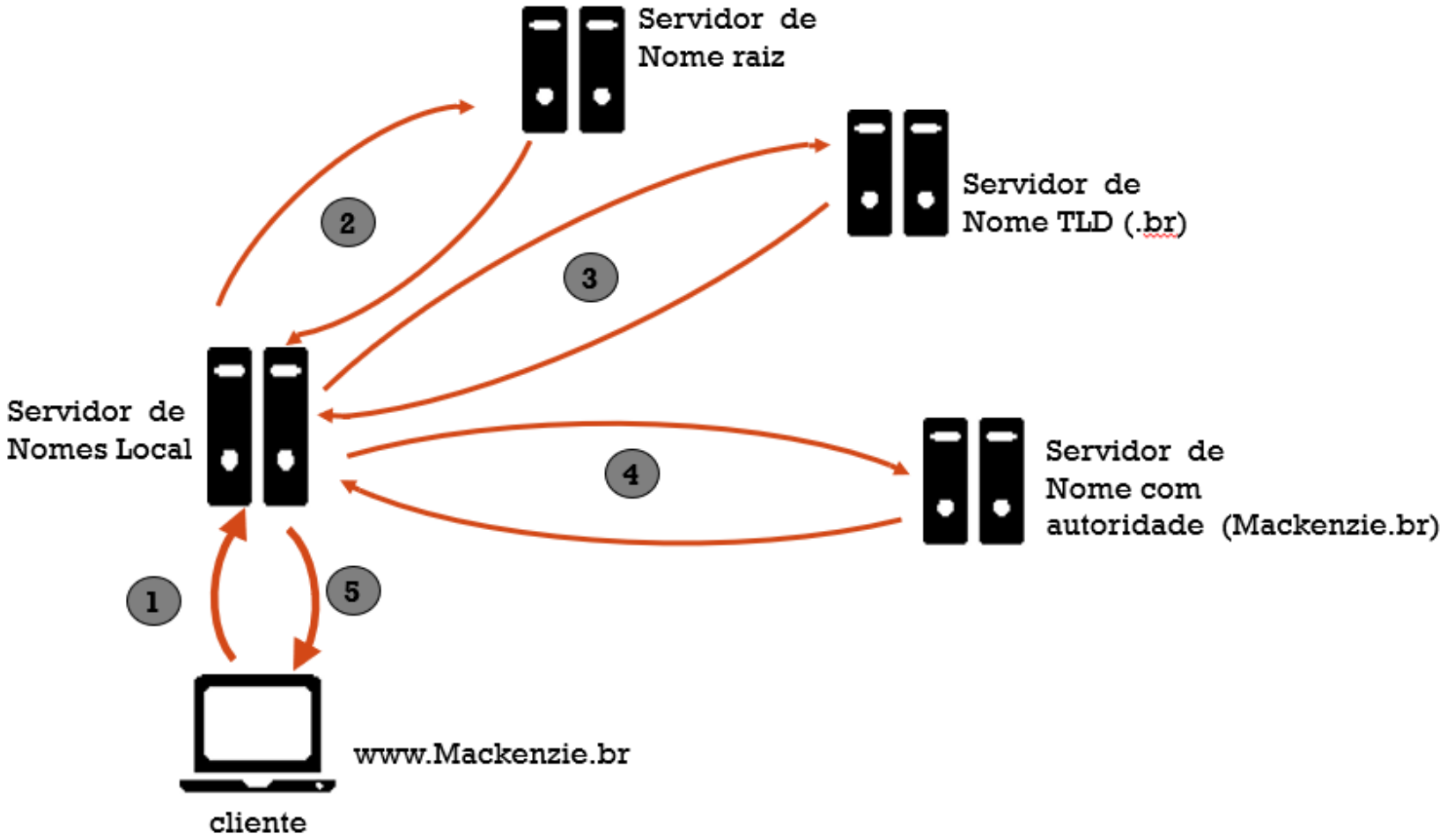
Figura 5 – Espaço de nome de domínios



Na Figura 5, podemos observar que, no topo da hierarquia dos servidores DNS, temos os servidores de nome raiz que são os responsáveis por iniciar a resolução de endereço. Controlados pela ICANN (Corporação da internet sem fins lucrativos, responsável pela alocação de endereços IP e pelos servidores DNS), existem 13 servidores raiz originais, assim como algumas centenas de cópias espalhadas pelo mundo. Os servidores de nome raiz possuem uma relação de todos os servidores de Domínio de Alto Nível (Top-level domain-TLD).
Os servidores TLD são divididos em servidores genéricos (generic top-level domains – gTLD), responsáveis por domínios de alto nível, como os ".com", ".net", ".edu", e pelos servidores por códigos de país (country-code top-level domains – ccTLD) responsável pelos servidores associados a países, como os servidores ".br", ".pt", ".uk", ".fr".
No terceiro nível hierárquico, temos os servidores de nome com autoridade que são gerenciados por grandes empresas, como o Google, Universidades ou provedores de Internet. Os servidores de nome com autoridade são responsáveis por abrigar os registros com os endereços IP dos servidores Web de seu respectivo mantenedor.

As consultas à hierarquia de servidores DNS são realizadas por meio de um servidor intermediário, disponibilizado pelos ISPs, por grandes empresas ou por servidores de nomes locais. A Figura 6 apresenta um esquema de como a consulta de um nome é realizada a partir de um servidor local.

Figura 6 – Exemplo de consulta DNS



No exemplo da Figura 6, temos um cliente querendo acessar o site da Universidade Presbiteriana Mackenzie (www.mackenzie.br). Para obter o endereço IP do servidor web da Mackenzie, o resolvedor cliente enviará uma mensagem com a URL digitada no browser para o servidor de nomes local (1). Caso o servidor de nomes local, que também pode servir como cache, não possua armazenada em sua unidade de armazenamento informações sobre o domínio solicitado (www.mackenzie.br), ele iniciará o processo de resolução de nomes consultando o servidor raiz (2). O servidor raiz não possui autoridade sobre o domínio consultado, mas, por meio do sufixo ".br", ele retorna ao servidor de nomes local o endereço do servidor TLD responsável pelo domínio ".br".

O servidor de nomes local enviará uma mensagem ao servidor TLD responsável pelo domínio ".br" (3). O servidor TLD não possui autoridade sobre o domínio solicitado, mas retornará ao servidor de nomes local o endereço do servidor com autoridade para responder a essa solicitação.

Ao receber o endereço do servidor com autoridade para responder pelo domínio www.mackenzie.br , o servidor de nomes local enviará uma mensagem para esse servidor e obterá o endereço IP do servidor web da Mackenzie (4). Após a resolução DNS, o servidor local armazenará a informação em sua cache e encaminhará o endereço IP para o cliente.

- **Registro de recursos**

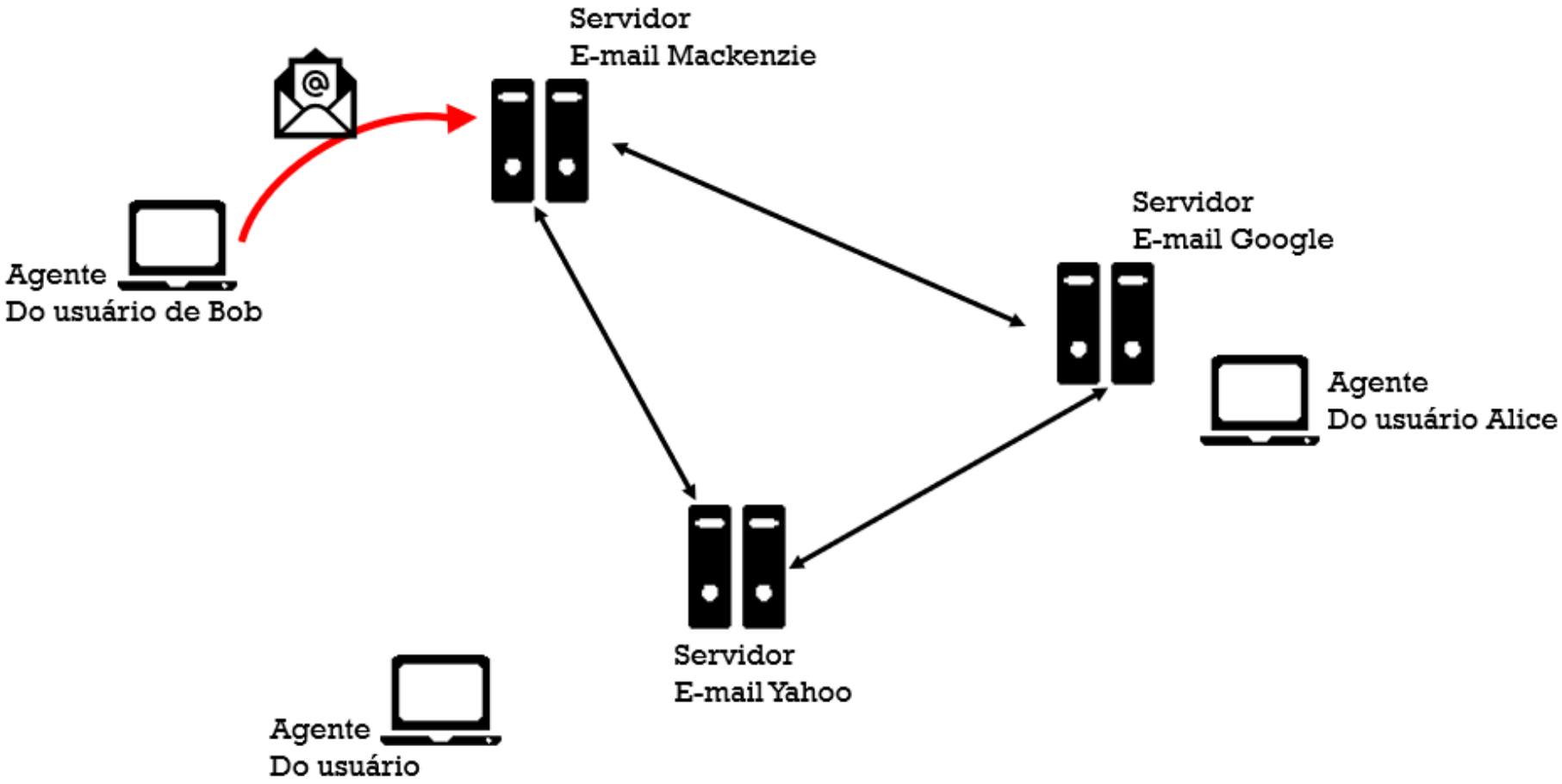
Todo servidor DNS possui uma base de dados com informações sobre os domínios sobre os quais ele possui autoridade. Essas informações contidas na base de dados dos servidores DNS são conhecidas como Registro de Recursos (Resource records – RR). O RR de cada domínio consiste em uma tupla contendo informações sobre o nome do domínio, o tipo do registro e o valor que serão consultados durante a resolução DNS. Detalhes sobre RR serão discutidos no recurso Professor Explica.

1.3. Correio eletrônico

Uma das primeiras aplicações criadas para troca de mensagens na rede, o correio eletrônico (e-mail), é certamente uma das aplicações mais importantes e mais utilizadas da Internet (Kurose, 2013).

Criado para ser a versão digital dos serviços de correio tradicionais, ao contrário de outros serviços de rede, o correio eletrônico realiza uma comunicação cliente-servidor assíncrona ou de mão única (Forouzan, 2010), ou seja, quando a pessoal “A” envia uma mensagem de correio eletrônico para a pessoa “B”, a pessoa “B” só receberá a mensagem quando consultar seu correio eletrônico e pode ou não responder a essa mensagem. Para que isso seja possível, a troca de mensagens de correio eletrônico (e-mail) é realizada por uma troca de mensagens entre diferentes servidores de correio eletrônico, conforme ilustrado na Figura 7.

Figura 7 – Diagrama ilustrativo de sistema de correio eletrônico da Internet



Na Figura 7, o usuário Bob possui o e-mail bob@mackenzista.br e deseja enviar um e-mail para Alice (alice@gmail.com). Para isso, Bob enviará a mensagem por meio de uma aplicação de correio eletrônico (agente do usuário) que encaminhará a mensagem até o servidor de correio eletrônico do Bob (servidor de e-mail da Mackenzie). O servidor de correio eletrônico do Bob, ao identificar que a mensagem tem como destino outro servidor de correio eletrônico, encaminhará a mensagem do Bob para o servidor de correio eletrônico da Alice (servidor de e-mail do Google). A mensagem enviada por Bob será armazenada no servidor de e-mail da Alice até ela acessar seu servidor por meio de seu agente do usuário.

Outra particularidade do serviço do correio eletrônico é que o envio e o recebimento de mensagens de correio são feitos por diferentes protocolos, em que o protocolo SMTP (Simple Mail Transfer Protocol) é responsável pelo envio de mensagens entre o agente do usuário do cliente e seu servidor de e-mail, assim como é responsável pela troca de mensagens entre os servidores de e-mail. Para acessar ou receber as

mensagens armazenadas no servidor de e-mail, o agente do usuário deverá usar os protocolos POP (Post Office Protocol), IMAP (Internet Mail Acess Protocol) ou o próprio HTTP por meio do Webmail.
Discutiremos um pouco mais sobre os protocolos de e-mail no recurso Professor Explica.

2. REFERÊNCIAS

COMER, D. Redes de Computadores e Internet. 2. ed. São Paulo: Porto Alegre, 2016.

FOROUZAN, B. A. Comunicação de dados e Redes de computadores. 3. ed. Porto Alegre: Bookman, 2010.

KUROSE, J. F.; ROSS, K. W. Redes de computadores e a internet: uma abordagem top-down. 6. ed. São Paulo: Pearson, 2013 | Biblioteca Virtual Universitária — Pearson.

TANENBAUM, A. Redes de computadores. 5. ed. São Paulo: Pearson, 2010 | Biblioteca Virtual Universitária — Pearson.

COMER, D. Redes de Computadores e Internet. 2. ed. São Paulo: Porto Alegre, 2016.

FOROUZAN, B. A. Comunicação de dados e Redes de computadores. 3. ed. Porto Alegre: Bookman, 2010.

KUROSE, J. F.; ROSS, K. W. Redes de computadores e a internet: uma abordagem top-down. 6. ed. São Paulo: Pearson, 2013 | Biblioteca Virtual Universitária — Pearson.

TANENBAUM, A. Redes de computadores. 5. ed. São Paulo: Pearson, 2010 | Biblioteca Virtual Universitária — Pearson.