



# Protocolos de Aplicação da Internet

Principais protocolos de aplicação da internet, seus conceitos, funcionamento básico, e ações para o monitoramento e a solução de problemas.

Prof. Fernando Diniz Hämmerli

### Propósito

Conhecer os principais protocolos de aplicação da internet é um requisito essencial para os profissionais de Tecnologia da Informação (TI). Para um desenvolvedor ou responsável por infraestrutura de servidores, é importante compreender como os protocolos operam para que os serviços oferecidos atendam às necessidades dos usuários. As rotinas de troubleshooting para a identificação e correção de problemas também são parte importante das atribuições desses profissionais.

### Preparação

Antes de iniciar este conteúdo, recomendamos possuir acesso à internet e ter o software Packet Tracer, da Cisco, instalado em seu computador para a realização das atividades propostas.

### Objetivos

- Descrever os protocolos de aplicação HTTP, FTP e TELNET.
- Descrever os principais protocolos de aplicação relacionados ao serviço de correio eletrônico.
- Descrever os protocolos de aplicação DNS, DHCP e SNMP.
- Reconhecer ferramentas e técnicas para monitorar o funcionamento dos protocolos estudados, com interpretação dos resultados.

### Introdução

Neste conteúdo, conheceremos os principais protocolos de aplicação utilizados na internet, responsáveis por grande parte do tráfego existente na rede.

Compreenderemos a necessidade que motivou a criação desses protocolos, como eles funcionam e vários aspectos práticos que a maioria dos usuários da internet desconhece por completo. Para os profissionais de TI, no entanto, são conhecimentos fundamentais, pois permitem a manutenção e evolução dos serviços existentes.

Antes de começar, assista ao vídeo a seguir, você entenderá a importância da camada de aplicação na internet e a evolução história.

Agora, antes de começarmos, assista ao vídeo e entenda a importância da camada de aplicação na internet e a evolução história.



#### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

## O protocolo HTTP

### Desvendando o protocolo HTTP

Neste vídeo, exploramos o mundo do Protocolo HTTP, o alicerce da web moderna. Descubra como o HTTP surgiu no início da internet e evoluiu para facilitar a troca de informações. Aprenda como ele trabalha com TCP e permite a transmissão de texto, imagens, vídeos e mais. Um mergulho nos bastidores da internet que todos devem conhecer!



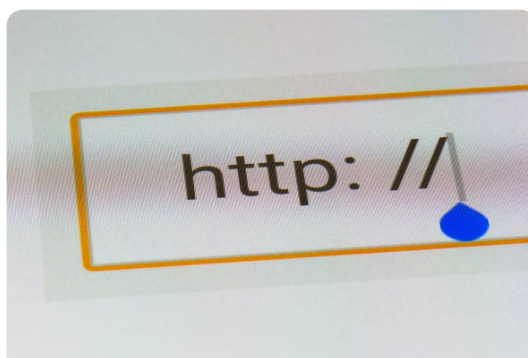
#### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Quando a internet começou a se popularizar, no final dos anos 1990, era comum ver programas de televisão e empresas divulgando seus sites. Eles sempre vinham em um formato estranho para quem não era da área técnica, mas se popularizaram rápido. Era sempre algo como “http://meusite.com.br”.

O que quase ninguém sabia é que esse formato se chama **URL**, ou Localizador Uniforme de Recursos, em tradução livre. Na prática, é uma frase compreensível para pessoas e máquinas que permite indicar o recurso que desejamos acessar e como ele deve ser acessado. Ainda neste módulo voltaremos a esse assunto.

O URL desse exemplo começa com “http://”, que indica ao seu programa navegador (Chrome, Firefox etc.) que o acesso deverá ser feito utilizando-se o protocolo de aplicação HTTP. E é por ele que começaremos os nossos estudos.



HTTP.

O protocolo HTTP foi criado no início dos anos 1990, como parte de um esforço para simplificar a troca de informações pela internet.

Naquele momento, a internet era uma rede puramente acadêmica e seu uso não era fácil. Na verdade, não se parecia em nada com a experiência que temos atualmente. A troca de informações era feita com serviços de poucos recursos e exigia dos usuários algum conhecimento técnico. Os próprios computadores da época com capacidade para acessar a internet eram pouco amigáveis.

Esse esforço rendeu um conjunto de soluções e ferramentas. Foi criada uma linguagem para a formatação de textos e com a possibilidade de navegação entre conteúdos, o chamado **hipertexto**. Essa linguagem recebeu o nome de **HTML**, está em contínua evolução e praticamente todos os sites se baseiam nela.

Também foi necessário criar programas para fornecer conteúdos, do lado do servidor (web servers) e para visualizar o conteúdo no computador cliente (browsers ou navegadores). Obviamente, se esses dois programas conversam, foi necessário definir as regras para essa conversa: como devem ser os pedidos e as respostas. Essas regras são o que chamamos de **protocolo**, e para esse conjunto de soluções foi criado o protocolo **HTTP**.

## HTTP

HTTP significa Hypertext Transfer Protocol, ou Protocolo de Transferência de Hipertexto, em tradução livre. Em 1997, foi publicada a versão 1.1 do protocolo HTTP, e ele funcionou tão bem que somente em 2015 foi lançada a versão 2.0, visando a atender demandas por sites mais interativos, em uma internet cada vez mais rápida, porém mantendo compatibilidade com a versão 1.1. Na versão 1.1, o protocolo HTTP trabalha sobre o protocolo de transporte TCP (Transmission Control Protocol).

Uma característica importante do HTTP é que ele permite transmitir diversos tipos de conteúdo, aqui chamados de **objetos**. Eles podem ser:



Texto



Imagens



Vídeos



Arquivos em geral

O TCP oferece um serviço confiável fim a fim, providenciando a retransmissão de pacotes de dados perdidos ou recebidos com erro. Ainda se encarrega de ajustar o tráfego às condições da rede e das estações participantes. Assim, o protocolo HTTP e as aplicações ficam dispensados de tomar esses cuidados que o TCP proporciona.

A porta padronizada para o HTTP é a 80, sendo possível usar qualquer outra desde que explicitada no URL. A conexão é sempre iniciada pelo cliente (navegador web) ao servidor web.

## Interações do protocolo HTTP

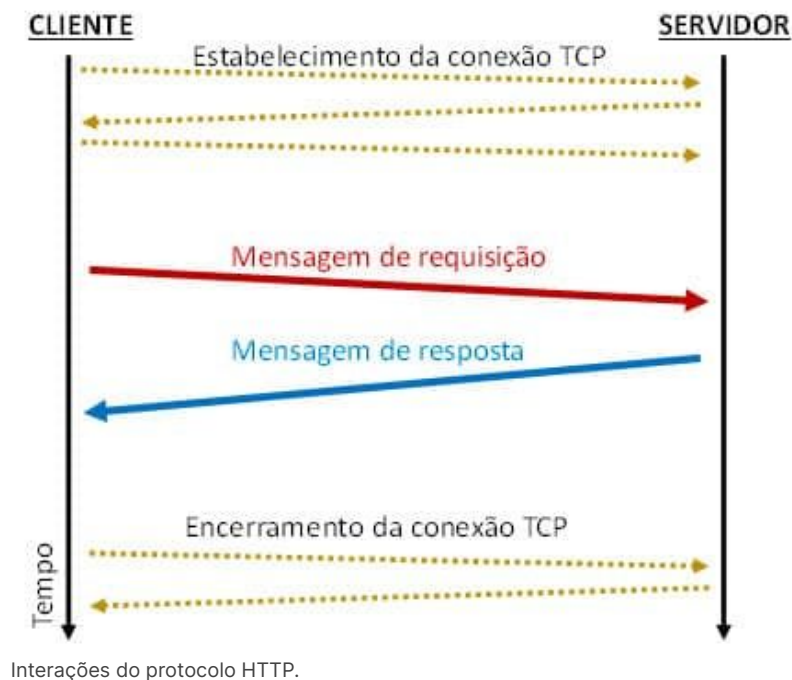
Neste vídeo, mergulhamos nas interações do Protocolo HTTP, desvendando como as mensagens de requisição e resposta funcionam. Entenda os elementos essenciais, como métodos, cabeçalhos e códigos de status, que guiam a comunicação entre cliente e servidor na web. Explore as diferenças entre conteúdo estático e dinâmico, e descubra como HTML, imagens, CSS e JavaScript se combinam para criar a experiência online que conhecemos. Um guia completo para compreender o coração da internet!



### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

As interações do protocolo HTTP seguem uma sequência: **mensagem de requisição** (pelo cliente) e **mensagem de resposta** (pelo servidor).



### Atenção

O processo de estabelecimento de conexão e encerramento da conexão TCP está fora do nosso escopo, portanto, as mensagens foram apresentadas para ilustrar o processo, mas não entraremos em detalhes de como ele é formado.

Uma característica importante do HTTP é que as requisições e respostas são escritas como um texto comum, usando palavras e expressões do idioma inglês. Isso nos permite olhar a comunicação e entender de imediato o que está acontecendo. Em protocolos que não seguem essa filosofia, fica muito difícil analisar seu funcionamento sem a ajuda de ferramentas externas.

### As mensagens de requisição seguem o formato:

- Linha de Requisição
- Linhas de Cabeçalho
- Linha sem conteúdo
- Corpo da Entidade (nem sempre usado)

A linha de requisição sempre começa com um **método**, que indica a ação que está sendo requisitada. As linhas de cabeçalho podem ser incluídas para carregar informações adicionais na mensagem de requisição. O corpo da entidade é usado quando o cliente precisa enviar dados ao servidor – por exemplo, quando preenchemos um formulário e esses dados estão sendo enviados para o servidor.

```
plain-text
```

```
GET / HTTP/1.1
```

```
Host: meusite.com.br
```

```
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:88.0) Gecko/20100101 Firefox/88.0
```

```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
```

```
Accept-Encoding: gzip, deflate
```

No exemplo anterior, temos na linha de requisição o método GET, usado quando o cliente deseja requisitar algum conteúdo. Ele é seguido pelo recurso que está sendo solicitado. No exemplo, o caractere “/” representa a raiz de um site, ou seja, a página principal. Mas poderia ter sido indicado um nome específico de conteúdo, como uma imagem ou outra página em HTML (/imagem.jpg ou /pagina.html). A linha de requisição termina com a indicação da versão do protocolo HTTP (1.1) que está sendo usada pelo navegador.

O HTTP prevê outros métodos, além do GET. Um deles é o POST, usado quando o cliente precisa enviar dados para o servidor – como exemplificado anteriormente, no caso do preenchimento de um formulário.

Após a linha de requisição, as linhas que seguem são de cabeçalho e carregam informações adicionais à requisição. A primeira delas, muito importante, informa qual o nome do site que está sendo acessado.

Lembre-se de que um servidor web pode servir a inúmeros sites simultaneamente e usando um único endereço IP.

Nas demais linhas, o cliente informa qual o programa (agente de usuário) está sendo usado (versão do navegador, sistema operacional), os tipos de conteúdo esperados e as codificações aceitas.

Para cada requisição recebida, o servidor enviará uma mensagem de resposta. As mensagens de **resposta** seguem o formato:

- Linha de Estado
- Linhas de Cabeçalho
- Linha sem conteúdo
- Corpo da Entidade (dados/conteúdo requisitados)

Observe o exemplo:

```
plain-text

HTTP/1.1 200 OK
Date: Thu, 06 May 2021 17:54:22 GMT
Server: Apache
Last-Modified: Tue, 04 May 2021 14:22:05 GMT
Content-Length: 149495
Content-Type: text/html; charset=UTF-8
(linha sem conteúdo) / Separa o cabeçalho do conteúdo
CONTEÚDO DA RESPOSTA
No caso deste exemplo seria um texto no formato HTML, informação declarada no cabeçalho
'Content-Type' acima.
```

A primeira linha (linha de estado) fornece três informações importantes:

- A versão do protocolo HTTP usada.
- O código de estado (status code).
- A frase explicativa (reason phrase).

O código de estado é sempre um número de três dígitos, que indica o resultado obtido para a requisição. A frase explicativa é um texto voltado para pessoas contendo informações sobre o resultado obtido para a requisição.

## Códigos de estado

O primeiro dígito do código de estado indica o tipo de resultado obtido, como indicado a seguir:

1XX

---

Informação. Requisição recebida, ainda sendo processada.

2XX

---

Sucesso. A requisição foi corretamente recebida e processada.

3XX

---

Redirecionamento. O cliente deverá tomar outra ação para concluir a requisição.

4XX

---

Erro do cliente. A requisição é inválida ou não pode ser atendida.

5XX

---

Erro do servidor. O servidor falhou ao processar uma requisição válida.

Alguns códigos de estado bem conhecidos são:

200

---

OK. Indicando que o processamento da requisição foi bem-sucedido.

404

---

Recurso não encontrado. Quando o cliente solicita um recurso que não existe no servidor.

500

---

Erro interno do servidor. Quando o servidor enfrenta um erro crítico ao processar uma requisição.

## Entendendo o que há por trás do conteúdo

Quando fazemos uma requisição ao servidor por um recurso, precisamos entender o que pode acontecer em seguida. As duas situações mais comuns são:



### Conteúdo estático

O conteúdo está em um arquivo armazenado no servidor, que será enviado ao cliente. Esse conteúdo é chamado de estático, pois ele não é criado ou formatado no momento da requisição. Para mudá-lo, um administrador deve modificar o arquivo no servidor. Nessa categoria estão páginas, imagens, arquivos de formato etc.

### Conteúdo dinâmico

Ao receber a requisição, o servidor executará um programa que irá gerar o conteúdo para o cliente. Esse conteúdo é chamado de dinâmico, pois é gerado na hora e pode variar conforme a requisição. É o caso de sites como webmail ou redes sociais, em que o conteúdo é personalizado para quem fez a requisição. Alguns exemplos de linguagens de programação e ambientes muito conhecidos para a execução dinâmica são o JAVA, PHP, ASP.NET, Python, entre outros.

Quando acessamos uma página, dificilmente apenas um recurso será baixado – na verdade, a maioria das páginas atualmente é composta por dezenas ou até centenas de recursos que precisam ser baixados para a montagem final da página.

O código HTML de uma página traz as instruções para a montagem dessa página na tela e o caminho (URL) para que o navegador baixe todos os elementos que serão necessários, como imagens, animações, vídeos etc.

Ao analisar os conteúdos baixados pelo navegador para montar uma página, possivelmente vamos nos deparar com:

#### HTML

---

Uma página pode ser composta por diversos conteúdos HTML, que podem ter origem em vários sites. Por exemplo, quando acessamos uma página e vemos aqueles quadros com anúncios, provavelmente aquele conteúdo foi gerado em uma plataforma especializada e incorporado à página que estamos acessamos.

#### Imagens

---

Vários formatos de imagem podem ser usados, como GIF, JPG, PNG.

#### CSS

---

Trazem instruções de estilo para formatar a página que estamos acessando. Por meio delas, o web designer define fontes, cores, espaçamentos da página.

#### JavaScript

---

São lógicas de programação a serem executadas pelo navegador. Permitem criar páginas interativas e com comportamento amigável aos usuários.

## HTTP com Conexões Persistentes e Segurança HTTPS

Neste vídeo, exploramos como o Protocolo HTTP evoluiu com conexões persistentes para otimizar o carregamento de recursos da web. Descubra como esse modelo reduz o tempo gasto na inicialização de conexões TCP e melhora a eficiência. Além disso, aprenda sobre a importância do HTTPS na segurança dos dados, como ele utiliza criptografia para proteger informações sensíveis durante a comunicação na internet.



### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

### Atividade

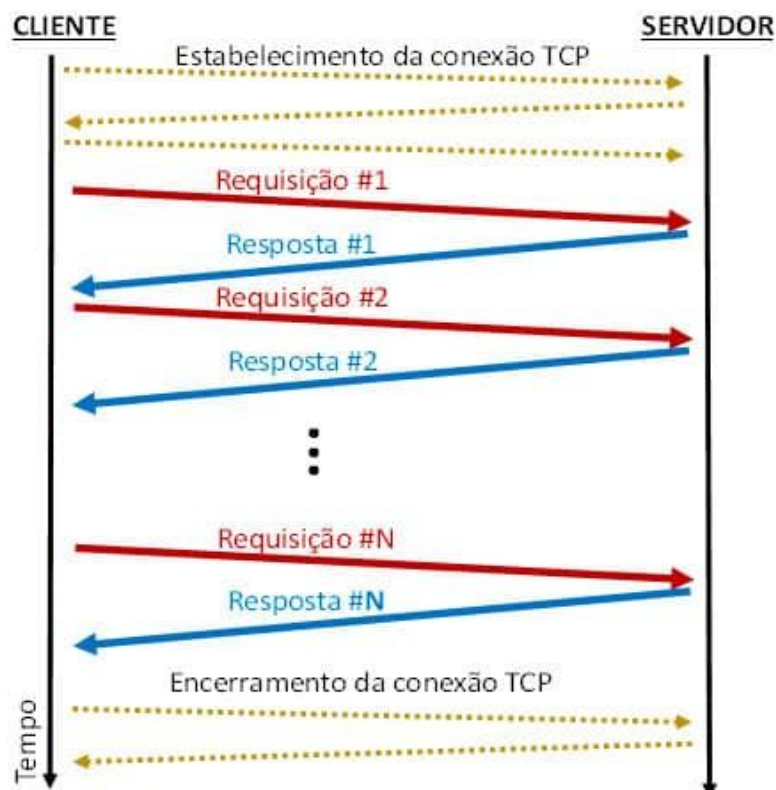
No início deste módulo, apresentamos um diagrama da interação entre cliente e servidor, em que era estabelecida uma conexão TCP, a requisição era feita, e a conexão era encerrada após o recebimento da resposta. Você conseguiu identificar o problema desse modelo?

### Chave de resposta

O problema é o tempo gasto para iniciar e encerrar conexões TCP quando precisarmos fazer muitas requisições para obter todos os recursos de uma página.

Por exemplo, imagine que o código HTML de uma página faça referência a 20 outros recursos, como imagens. No total, o navegador precisará fazer 21 conexões TCP ao servidor e requisitar apenas um recurso em cada conexão. Esse modo de operação, chamado de não persistente, acaba oferecendo uma baixa eficiência.

Para aumentar o desempenho, reduzindo o tempo total necessário para baixar os recursos, o protocolo HTTP prevê o modo de operação persistente. Nele, a conexão TCP permanece ativa após uma resposta, estando pronta para transferir uma nova requisição imediatamente.



A conexão TCP será encerrada quando o cliente não tiver mais requisições a fazer, ou pelo servidor, após um período sem atividade.

## URL – Uniform Resource Locator

Iniciamos este módulo falando sobre o URL, aquela frase amigável para pessoas, contendo as informações necessárias para um navegador buscar um recurso. Um URL completo, com todos os dados possíveis, teria um formato assim:

```
protocolo://hospedeiro:porta/caminho/recurso
```

A seguir, veremos o que significa cada um dos elementos:

### Protocolo

É o protocolo de aplicação que deverá ser usado para realizar o acesso ao servidor hospedeiro. Por exemplo: http.

### Hospedeiro

É o endereço do servidor a ser acessado. Pode ser um endereço IP (IPv4 ou IPv6) ou um nome. No último caso, o navegador deverá realizar a tradução do nome pelo serviço de DNS, antes de iniciar a conexão.

### Porta

É um campo opcional, indicando explicitamente a porta (protocolo TCP) que deverá ser usada. Quando não especificado, é usada a porta padrão do protocolo. No caso do HTTP, a porta padrão é a 80.

### Caminho e recurso

É a localização do recurso no servidor, semelhante a uma estrutura de diretórios, facilitando a organização do conteúdo. O caminho pode ser composto por nenhum ou até por vários nomes. Veja os exemplos:

- <http://www.meusite.com.br/imagens/equipamentos/telefone.jpg>
- <http://www.meusite.com.br/telefone.jpg>

Nesses casos, não foi especificada a porta, então, foi usada a porta 80 como padrão. No exemplo a seguir, o URL especifica uma porta diferente.

- <http://www.meusite.com.br:8080/documentos/contrato.pdf>

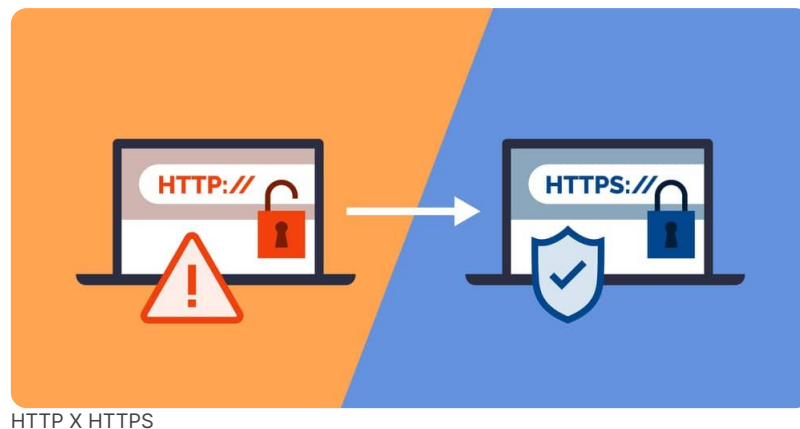
# HTTPS

Provavelmente, você está sentindo falta neste texto do HTTPS, e qual a relação dele com o protocolo HTTP.

Não é tarefa do HTTP garantir a segurança dos dados trafegados. Qualquer pessoa que consiga capturar os pacotes de uma conversa HTTP terá acesso completo aos dados trafegados. Conteúdo de páginas, formulários preenchidos, senhas, cartões de crédito, tudo é trafegado sem qualquer proteção, sem garantir a confidencialidade.

Igualmente, o protocolo HTTP não tem recursos para impedir um ataque em que o conteúdo da comunicação é maliciosamente modificado durante seu trânsito, entre origem e destino.

Para resolver essas questões e conferir segurança ao protocolo HTTP, foi criado, em 1994, o HTTPS. Em vez de criar um protocolo, o HTTPS foi definido como **uma extensão do HTTP**, contendo mecanismos para garantir a confidencialidade e a integridade dos dados transmitidos.

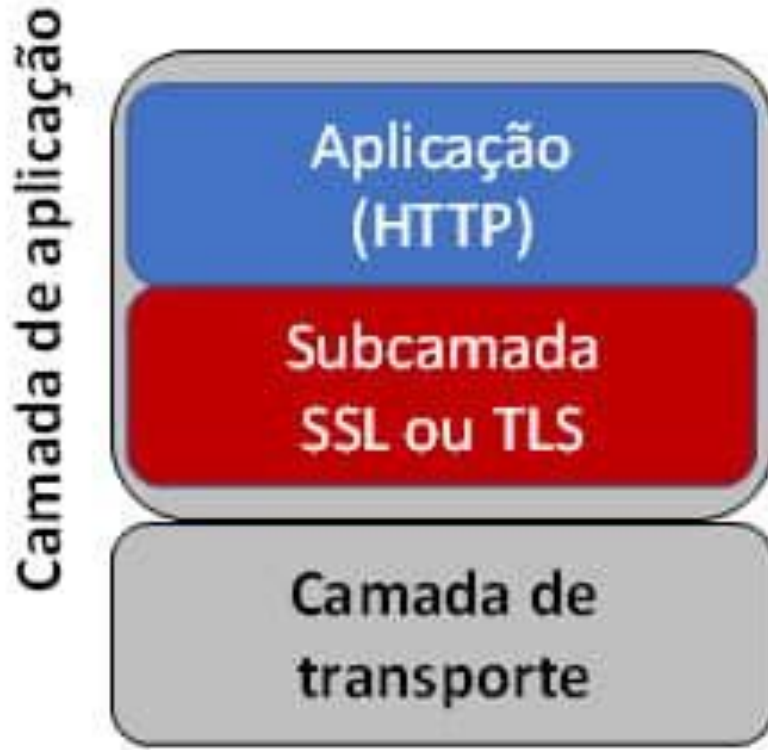


HTTP X HTTPS

## Mas, como o HTTPS funciona?

A segurança é obtida utilizando mecanismos de criptografia simétrica e assimétrica, e hashing para assinatura digital. O HTTPS também utiliza certificados digitais, assinados por autoridades certificadoras externas. Assim, é possível também impedir que alguém crie um site falso usando o nome ou endereço de alguma empresa ou entidade.

O HTTPS funciona com uma subcamada, abaixo do HTTP, em que todas as informações são protegidas antes de serem enviadas para a camada de transporte; no receptor, a operação inversa é feita nos dados enviados pela camada de transporte. Essa subcamada é composta por um mecanismo SSL (Secure Socker Layer) ou, mais recentemente, TLS (Transport Layer Security).



Modelo simplificado do protocolo HTTPS.

Fora da subcamada de segurança, a troca de mensagens no HTTPS é exatamente a mesma do HTTP, seguindo as mesmas regras.

## Protocolo FTP

### Explorando o protocolo FTP

Neste vídeo, adentramos o mundo do Protocolo FTP, um dos principais meios de transferência de arquivos na internet. Descubra como o FTP difere do HTTP e seu uso de conexões persistentes para navegar, listar e transferir arquivos. Explore os modos de operação ativo e passivo e saiba como superar desafios com firewalls.



#### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Antes do HTTP, o principal protocolo para a transferência de arquivos na internet era o FTP, mas ainda muito utilizado. Por ser dedicado à transferência de arquivos, seu funcionamento é diferente em comparação com o HTTP, porém também utiliza o protocolo de transporte TCP.

O FTP trabalha com uma conexão persistente, a conexão de controle, pela qual um usuário interage com o servidor, enviando comandos para navegação entre diretórios, lista seu conteúdo e inicia transferências de arquivo – GET (do servidor para o cliente) e PUT (do cliente para o servidor).

A porta padrão do servidor FTP para a conexão de controle é a 21.

## Conexões do protocolo FTP

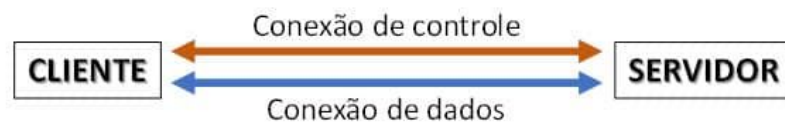
Ao estabelecer a conexão de controle, o servidor envia uma saudação com uma mensagem que pode ser personalizada pelo administrador. Em seguida, é necessário se autenticar por meio de login e senha.



### Curiosidade

Quando o servidor tem conteúdo público, seu administrador pode optar por uma autenticação anônima, assim qualquer pessoa pode acessar o servidor. Isso é muito usado, por exemplo, em servidores FTP onde são disponibilizadas distribuições do Linux para o público em geral.

Uma vez autenticado o usuário, a conexão de controle permanece ativa. Para qualquer transferência de arquivos, uma nova conexão será iniciada, a **conexão de dados**, que será encerrada imediatamente após o término da transmissão.



Conexões utilizadas pelo protocolo FTP.

Uma diferença importante na operação dos protocolos FTP e HTTP é apresentada a seguir:

#### FTP

Utiliza o **controle fora da banda**, pois separa em conexões distintas os comandos e transferências.



#### HTTP

Utiliza o **controle na banda**, pois usa a mesma conexão para comandos e transferências.

## Modos de operação – ativo e passivo

Um ponto importante e que todo profissional de TI deve conhecer, são os dois **modos** para estabelecer as conexões de dados pelo FTP:

#### Modo ativo

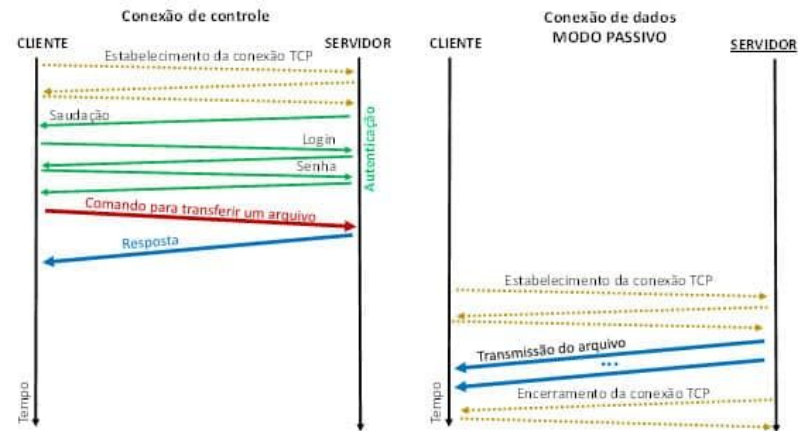
Neste modo, a conexão de dados é iniciada **pelo servidor** ao cliente, com porta de origem 20. Ou seja, para iniciar uma transferência, o cliente deve abrir uma porta e aguardar que o servidor se conecte a ela. O grande problema desse modo é que atualmente grande parte dos terminais acessa a internet com tradução de endereço (NAT – Network Address Translation) ou está em redes protegidas por firewalls, que impedem conexões externas.



#### Modo passivo

Nos casos de dificuldade de uso do modo ativo, a solução é utilizar o modo passivo, em que a conexão é iniciada pelo cliente para o servidor em uma porta diferente da porta de controle. Para usar o modo passivo, o cliente envia ao servidor, na conexão de controle, o comando PASV e recebe como resposta um endereço IP e porta na qual a conexão deverá ser iniciada.

A figura a seguir ilustra a requisição de um arquivo através do protocolo FTP, operando em modo passivo.



Conexões do protocolo FTP operando em modo passivo.



### Atenção

O modo passivo não funcionará se o cliente estiver em uma rede protegida por firewall que limite as portas nas quais as conexões são permitidas. Para resolver esse problema, sem precisar liberar a saída para qualquer porta, a maioria dos firewalls conta com módulos que monitoram as conexões FTP e identificam a negociação para uma conexão passiva. Assim, eles conseguem liberar, com precisão cirúrgica, as conexões de dados.

## Protocolo TELNET

### TELNET: Explorando o protocolo de acesso remoto

Neste vídeo, mergulhamos no Protocolo TELNET, uma ferramenta pioneira que permite o acesso remoto a terminais via internet. Descubra como o TELNET opera, sua interação com o servidor e seu papel crucial em sistemas baseados em texto. Aprenda sobre suas limitações de segurança e alternativas modernas.



#### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

O principal objetivo do protocolo de aplicação TELNET é permitir o acesso remoto a terminais através da internet. Por meio dele, um usuário pode se conectar a um computador remotamente e ter a mesma experiência que utilizando o terminal desse computador presencialmente.

O TELNET é voltado apenas para terminais baseados em texto, ou seja, transferindo somente dados de caracteres.

O TELNET não permite, por exemplo, acesso a um ambiente gráfico ou o uso de mouse para interação. Essa característica tornou o TELNET uma opção ideal para o acesso a sistemas baseados em texto como o shell, estações baseadas em UNIX e para consoles de gerência de equipamentos de rede, como roteadores, switches etc.

### Interação com o servidor TELNET

O TELNET utiliza o protocolo de transporte TCP, que garante a confiabilidade na entrega dos pacotes.

O TELNET funciona da seguinte forma:

- Quando um cliente TELNET se conecta a um servidor, é criado um canal de comunicação bidirecional, em que ambos podem enviar e receber dados.
- Como o objetivo do TELNET é oferecer ao usuário uma experiência semelhante ao uso presencial, cada vez que o usuário pressiona uma tecla essa informação é imediatamente enviada ao servidor.
- O cliente não aguarda a digitação de um comando completo para enviá-lo. Na verdade, o cliente não tem conhecimento sobre os comandos digitados ou suas estruturas. Por outro lado, cada informação recebida pelo cliente é imediatamente exibida no terminal do usuário.
- O cliente TELNET sequer exibe no seu próprio terminal as teclas digitadas pelo usuário. Para que isso aconteça, o servidor deverá enviar de volta o texto digitado para o cliente, uma operação chamada ECO. Assim, o servidor tem o controle sobre o que será exibido no terminal do cliente.



#### Atenção

Uma situação em que esse comportamento é importante é a digitação das credenciais do usuário. A identificação (login) do usuário é ecoada pelo servidor e aparece na tela, porém a senha não deve ser exibida. Quando o usuário digita a senha, o servidor recebe cada caractere digitado, mas não o envia de volta. Opcionalmente, o servidor poderia ser configurado para retornar sempre um caractere coringa, como um asterisco (\*), apenas indicando que recebeu um caractere, sem revelar qual.

O protocolo TELNET não possui mecanismos de autenticação de usuário. Qualquer solicitação de identificação ou senha é transmitida e respondida como um texto comum. Muitos sistemas, apesar de não recomendado, permitiam o acesso ao seu terminal sem autenticação. Nesses casos, o terminal de comandos do servidor estava imediatamente disponível após a conexão do usuário.

Exemplo de acesso a um terminal usando o TELNET. O trecho das primeiras três linhas indica o processo de estabelecimento de conexão. Após estabelecido, os textos das demais linhas indicam o texto enviado pelo servidor. Os textos “**admin**” (5ª linha) e “**1s**” (8ª linha) indicam os textos digitados pelo cliente.



```
plain-text

$ telnet 192.168.30.17
Trying 192.168.30.17...
Escape character is '^]'.
Acesso restrito a administradores do sistema.

servidor login: admin
Password:
Welcome to Ubuntu 20.04 LTS (GNU/Linux 4.4.0-19041)

admin@servidor:~$ ls
arquivo1 arquivo2 arquivo3 arquivo4
admin@servidor:~$
```

A primeira linha mostra o cliente TELNET sendo executado com a indicação do endereço IP do servidor a ser conectado. O cliente inicia uma tentativa de conexão TCP à porta 23 do servidor. O servidor aceita a conexão e envia uma saudação, opcional e configurável por seu administrador.

Imediatamente, o servidor inicia o processo de autenticação, solicitando a digitação da identificação (login) do usuário e, logo em seguida, da sua senha. Repare que a senha, ao contrário do login, não é exibida no terminal.

Como a autenticação foi bem-sucedida, o servidor disponibiliza o terminal e o Prompt de comandos, no qual o usuário digita o comando "ls" e recebe como resposta a lista de arquivos no diretório atual.

O TELNET usa a porta 23 como padrão.

## Segurança

Atualmente, o protocolo TELNET está em desuso e é extremamente desaconselhável utilizá-lo na internet. A razão para isso é a falta de segurança. Toda a comunicação é transmitida em texto claro, ou seja, sem qualquer tipo de criptografia que impeça a captura de seus dados, inclusive senhas. A alternativa para acesso remoto atualmente é o protocolo SSH (Secure Shell Protocol), que possui criptografia e autenticação integrada.



### Atenção

Sempre utilize senhas fortes para evitar que ataques de força bruta tenham sucesso.

## Verificando o aprendizado

Questão 1

O URL é uma frase para localização de recursos, de fácil compreensão para pessoas e máquinas, com os elementos necessários para que o recurso referenciado seja obtido. Alguns desses elementos são obrigatórios, enquanto outros são opcionais no URL, apesar de continuarem sendo essenciais para a localização do recurso. Nesse contexto, qual deve ser a ação de um navegador web quando a informação de porta é omitida de URL iniciada com "http://"?

A

Como se trata de informação explícita obrigatória no URL, o navegador retornará um erro ao usuário indicando a impossibilidade de acesso ao recurso.

B

Como o protocolo de aplicação HTTP não utiliza número de porta, essa é uma informação dispensável no URL para esse protocolo.

C

O navegador deverá utilizar a porta padrão estabelecida para o protocolo de aplicação indicado no início do URL.

D

Como já é estabelecida a porta 80 para o protocolo HTTP, essa informação não deve constar no URL e, se existir, será ignorada pelo navegador web.

E

Como o número de porta é informação inerente ao protocolo de transporte, e não ao de aplicação HTTP, esse elemento é dispensável no URL.



A alternativa C está correta.

A RFC 1738, que estabelece o URL, determina que, se o número da porta não for explicitado, a porta padrão para o protocolo deve ser utilizada. No caso do protocolo HTTP, é a porta 80, enquanto para o HTTPS é a 443.

Questão 2

Uma das características do protocolo FTP é o controle fora da banda, que permite iniciar uma segunda transferência de arquivo mesmo enquanto a primeira ainda está em curso. Esse comportamento é possível, pois

A

os comandos são enviados por uma conexão exclusiva e, para cada transferência, uma nova conexão é iniciada.

B

os arquivos são transferidos em blocos e, após cada um, o cliente solicita o próximo bloco, podendo suspender uma transferência e iniciar outra.

C

o protocolo FTP utiliza o protocolo de transporte UDP, que permite o envio simultâneo de diversos fluxos de transmissão.

D

o protocolo FTP se beneficia da multiplexação de processos oferecida pelo socket de um sistema operacional, que compartilha a infraestrutura de rede local.

E

o protocolo FTP utiliza uma única conexão por onde os fluxos de comandos e dados são transmitidos. Por meio de identificação única, os fluxos são separados no cliente e servidor.



A alternativa A está correta.

O protocolo FTP utiliza uma conexão TCP chamada de conexão de controle, por onde o usuário inicia sua sessão, realiza a autenticação e envia comandos para navegação em diretórios, para enviar ou buscar arquivos. Para realizar a transferência do arquivo, uma nova conexão TCP é criada, exclusiva para esse fim. A conexão de controle permanece ativa e disponível para que o cliente envie novos comandos ao servidor.

# Protocolos de correio eletrônico

No vídeo a seguir, você verá o funcionamento do serviço de correio eletrônico e os protocolos utilizados para que uma mensagem possa sair do remetente e chegar ao destinatário.



### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Um dos serviços mais antigos e mais usados na internet é o correio eletrônico, popularmente conhecido como e-mail, que recebeu esse nome como uma analogia direta ao serviço tradicional de correio postal.

O correio eletrônico é baseado em uma estrutura simples de serviços e protocolos, responsáveis pelas seguintes tarefas:

### Recepção >> Enfileiramento >> Encaminhamento >> Armazenamento >> Resgate de mensagens

Neste módulo, estudaremos esses elementos e como se relacionam, e conheceremos os protocolos que tornam tudo isso possível.

Os protocolos de correio eletrônico são usados pelos clientes de correio, instalados no terminal do usuário, para se comunicarem com os servidores de correio eletrônico.

Os principais sistemas operacionais possuem clientes nativos, geralmente mais limitados em recursos. Em ambientes corporativos, clientes mais sofisticados costumam ser preferidos, como o Outlook ou o Thunderbird. Do lado do servidor, processos dedicados realizam as tarefas de transmissão, armazenamento e acesso aos e-mails.



## Protocolos SMTP

Vamos iniciar com um exemplo simples, em que João, da empresa X, envia um e-mail para Maria, da empresa Y. Cada um deles tem o seu próprio endereço de correio eletrônico:

- [joao@empresax.com](mailto:joao@empresax.com)
- [maria@empresay.com](mailto:maria@empresay.com)

O próprio formato do endereço de e-mail já nos mostra a estruturação hierárquica do serviço de correio. O endereço é composto de duas partes, separadas pelo caractere @. A segunda parte corresponde ao domínio, enquanto a primeira é a parte local, a identificação do usuário dentro do domínio.



Exemplo de endereço de e-mail

Para fazer o envio, João usa um cliente de correio eletrônico, em que redige o texto, digita o endereço de Maria como destino e dá o comando para enviar. A sequência mais simples de passos ocorre do seguinte modo:

#### Cliente se conecta ao servidor

O cliente de correio do João se conectará ao servidor de correio da sua própria empresa (empresa X). Esse servidor é o responsável por receber as mensagens enviadas pelos endereços da empresa.

#### Servidor recebe a mensagem

O servidor da empresa X receberá a mensagem e a armazenará em uma área temporária. Será feita uma análise do destinatário da mensagem. O servidor da empresa X determinará o endereço do servidor de destino, ou seja, da empresa Y.

#### Conexão entre servidores

O servidor da empresa X se conectará ao servidor da empresa Y e enviará a mensagem a ele.

#### Servidor do destino recebe a mensagem

O servidor da empresa Y receberá a mensagem e a armazenará a mensagem na caixa de mensagens daquele usuário.

Essas etapas estão ilustradas a seguir.



Caminho percorrido por um e-mail enviado.

Repare que o último item é semelhante a uma caixa de correio física, como usado em muitos prédios. As mensagens recebidas são colocadas nessa caixa e, posteriormente, o destinatário precisará acessar sua caixa para receber o que está ali armazenado.

As etapas 2 e 4 ocorreram nos servidores como tarefas internas. Porém, nas etapas 1 e 3 ocorreu transmissão através da rede. E essas duas etapas têm em comum o protocolo de aplicação usado para a transmissão da mensagem: o **SMTP (Simple Mail Transport Protocol)**.

O SMTP atua como um agente de transmissão de mensagens (MTA – Mail Transfer Agent), tornando possível o envio da mensagem do remetente, até chegar à caixa postal do destinatário.

Vamos iniciar analisando as etapas 2 e 3, quando o servidor da empresa X envia a mensagem ao servidor da empresa Y.

Logo que o servidor da empresa X recebe a mensagem do cliente (João), ele identifica que o destinatário não é um usuário local daquele domínio. Se o destinatário fosse maria@empresax.com, o servidor apenas armazenaria a mensagem na caixa dessa usuária (obviamente, supondo que ela exista). Porém, o destino está localizado em outro domínio, o empresay.com.

**Então, o que precisa ser feito para que a conexão ocorra?**

Antes de se conectar ao servidor do destinatário, é preciso determinar qual o endereço IP desse servidor. Para isso, o servidor da empresa X buscará no serviço de DNS o registro do tipo MX do domínio empresay.com.

Uma vez que o endereço foi determinado, o servidor passa à etapa 3. Uma conexão SMTP é iniciada no endereço do servidor de destino.



#### Atenção

O protocolo SMTP utiliza o protocolo de transporte TCP, sua porta padrão é a 25 e emprega um modelo cliente-servidor, com a conexão TCP sempre iniciada pelo cliente. Portanto, teremos no processo de encaminhamento das mensagens o Cliente MTA e o Servidor MTA. Na etapa 3 desse exemplo, o servidor da empresa X (servidor X) assume o papel de CLIENTE MTA, iniciando a conexão para o SERVIDOR MTA da empresa (servidor Y).

## Mensagens do protocolo SMTP

Toda a conversa pelo protocolo SMTP, desde a negociação inicial até o término da transmissão da mensagem, será feita com caracteres de texto. Portanto, assim como vimos no HTTP, se o tráfego for capturado será possível observar com facilidade toda a comunicação.

Eis um exemplo (simplificado) de uma comunicação SMTP para o envio de uma mensagem:

plain-text

```
(01) S: 220 mailserver.empresay.com ESMTP Postfix
(02) C: EHLO mailserver.empresax.com
(03) S: 250-SIZE 1048576
(04) C: MAIL FROM:
(05) S: 250 Ok
(06) C: RCPT TO:
(07) S: 250 Ok
(08) C: DATA
(09) S: 354 End data with .
(10) C: From: "Joao Silva"
(11) C: To: "Maria Souza"
(12) C: Date: Tue, 18 May 2021 12:03:15 -0300
(13) C: Subject: Confirmação
(14) C:(linha sem conteúdo enviada pelo cliente, para separar cabeçalho do corpo da
mensagem)
(15) C: Olá, Maria.
(16) C: Confirmo que recebi seu pedido.
(17) C: Atenciosamente,
(18) C: João.
(19) C: .
(20) S: 250 Ok: queued as 1F8C5A549
(21) C: QUIT
(22) S: 221 Bye
```

As linhas iniciadas com "S:" indicam conteúdo enviado pelo servidor (servidor Y).

As linhas iniciadas com "C:" indicam o conteúdo enviado pelo cliente (servidor X).

Repare que as respostas do servidor são sempre iniciadas com um código numérico de três dígitos, o status code, semelhante ao HTTP. Resumidamente, se o código de for iniciado com:

2XX

---

Resposta positiva, a solicitação foi feita com sucesso.

3XX

---

A solicitação foi recebida com sucesso, porém, aguarda alguma pendência para ser concluída.

4XX

---

Erro ao processar a solicitação, por uma condição temporária. O cliente deverá repetir a solicitação posteriormente.

5XX

---

Erro ao processar a solicitação. O cliente **não** deve repetir a solicitação.

Os programas que usam o protocolo SMTP baseiam seu funcionamento somente nas respostas numéricas. O texto posterior serve para auxiliar no diagnóstico de problemas pelos administradores do sistema.

Vamos analisar a conversa SMTP do exemplo anterior:

1

Logo após o estabelecimento da conexão TCP, o servidor envia uma saudação, seguindo o código 220 que significa “estou pronto” (linha 1).

2

Em seguida, o cliente se identifica com o comando EHLO. O servidor responde com informações sobre opções ou recursos disponíveis. Nesse exemplo, foi informado somente o tamanho máximo de mensagem aceita pelo servidor. Se a mensagem a ser enviada fosse maior que esse limite, o cliente deveria interromper o envio e reportar ao remetente a impossibilidade de transmissão (linhas 2 e 3).

3

Continuando, o cliente declara o remetente (MAIL FROM) e os destinatários da mensagem, nesse caso, apenas um destinatário. Se fossem mais destinatários para aquele servidor (@empresay.com), o comando RCPT TO: seria repetido para cada um deles (linhas 4 a 7).

4

Em seguida, o cliente usa o comando DATA para começar o envio da mensagem. O servidor responde que ele deve começar a enviar o conteúdo e sinalizar o término, enviando uma linha apenas com um ponto (linhas 8 e 9).

5

A mensagem é enviada. Repare que ela é dividida em duas partes, separadas por uma linha sem conteúdo (linha 14). A primeira parte, é chamada de CABEÇALHO (linhas 10 a 13). É onde são declaradas as informações daquela mensagem, como remetente, destinatários, cópia aberta, assunto, data e hora do envio etc. A segunda parte é o corpo, que corresponde ao texto da mensagem (linhas 15 a 18).

6

A linha 19 (apenas com um caractere “ponto”) indica o término da mensagem ao servidor.



7

O servidor responde com código 250, para indicar operação bem-sucedida (linha 20).

8

Terminada a transmissão, o cliente envia um comando QUIT e o servidor encerra a conexão TCP (linhas 21 e 22).

O SMTP define que o trecho até a linha 9 é chamado de **envelope**, enquanto o restante é chamado de **conteúdo**.

Nesse exemplo, o RCPT TO: (linha 6) teve respostas com código 250, que indica sucesso da requisição. Se o destinatário estivesse com a caixa de mensagens lotada, o código seria um erro 450, e o cliente deveria tentar o envio posteriormente.

Se o destinatário não existisse, por exemplo, a resposta seria um código 550. Nesse caso, o servidor deveria descartar a mensagem e enviar uma notificação ao remetente. Nas notificações, o texto recebido com o código 550 costuma ser incluído, o que ajuda no diagnóstico do problema.



### Recomendação

Faça uma experiência: envie um e-mail para dois destinatários. Um deles deve ser válido (pode ser o seu próprio e-mail). O segundo deve ser inválido (coloque caracteres aleatórios e um domínio válido). O e-mail deverá chegar ao endereço válido e o remetente receberá uma resposta indicando a impossibilidade de enviar para o endereço inválido. Verifique se a resposta 5XX do SMTP foi incluída e o texto com informações sobre o erro.

## Necessidade de autenticação no protocolo SMTP

Já vimos a transmissão na etapa 3, entre os servidores. Porém, ainda falta falarmos da etapa 1, entre o cliente, no computador de João, e o servidor da empresa X. A conversa com o protocolo SMTP é praticamente a mesma, mas há uma consideração de segurança importante.

O servidor X recebe a mensagem enviada pelo cliente de João e a retransmite para o servidor de destino. Imagine que o servidor X seja plenamente acessível na internet para qualquer usuário, e não somente para quem realmente trabalha na empresa X. Esse servidor seria um alvo fácil para pessoas que desejam transmitir SPAM e para tentativas de golpe usando o servidor da empresa X.

Esse era o cenário da internet até meados dos anos 1990, quando o SPAM se tornou uma realidade. Antes liberados, a alternativa foi restringir o acesso aos servidores de SMTP a quem é efetivamente usuário daquele servidor.

Para isso, foram criadas extensões no protocolo SMTP para exigir a autenticação dos usuários. Alguns dos procedimentos de segurança estão listados a seguir:

- A autenticação é, quase sempre, realizada por meio de senha pessoal.
- O servidor somente retransmitirá uma mensagem para outro servidor se ela tiver sido recebida de um usuário devidamente autenticado.
- O cliente utiliza funções de hashing para enviar a senha ao servidor, impedindo que ela seja interceptada mesmo quando a comunicação não é criptografada.

## Uso de criptografia com o protocolo SMTP

Atualmente, a grande maioria das comunicações SMTP é feita com o uso de criptografia, protegendo toda a comunicação, inclusive o conteúdo das mensagens, de captura por terceiros.

Sobre isso, a escolha da porta a ser usada depende do tipo de conexão:

### Cliente-servidor

Nos dias atuais, é amplamente recomendado que se utilize a porta **587/TCP** para as conexões dos clientes de usuário aos servidores para a transmissão de mensagens, com o uso obrigatório de criptografia e autenticação. A porta 587 é conhecida como **submission port**.

### Servidor-servidor

Entre servidores, é usada somente a porta 25, e se ambos suportarem a camada de segurança TLS (Transport Layer Security), ela poderá oferecer criptografia e segurança por meio de certificados digitais.

## Cabeçalho Received nos e-mails

Todas as vezes em que uma mensagem é recebida e retransmitida por um servidor de correio, ele inclui uma linha no início do cabeçalho que começa com RECEIVED:. Esse cabeçalho oferece um registro do caminho percorrido pela mensagem, de servidor para servidor, permitindo rastreá-la. Nela, estão incluídas informações sobre cada vez que foi retransmitida, como servidor de origem e destino, data e hora.



### Dica

Geralmente, os programas clientes de e-mail (inclusive webmail) ocultam esses cabeçalhos. Para vê-los, procure por opções como “ver cabeçalho completo” ou “ver código fonte” ou “mostrar original”. O nome da opção varia conforme o programa.

## Protocolos POP3 e IMAP – resgatando mensagens no servidor

Até agora, vimos o protocolo SMTP funcionando como um carteiro que entrega uma carta na caixa de correio de uma casa. Porém, para essa carta chegar ao seu destinatário, ele deverá abrir essa caixa e resgatar as

cartas armazenadas. No correio eletrônico, essa lógica é a mesma e, para essa função, estudaremos dois protocolos de resgate de mensagens:

### POP (Post Office Protocol)

Porta padrão: 110

Versão mais recente (no momento da confecção deste material): POP3 (terceira versão)

### IMAP (Internet Message Access Protocol)

Porta padrão: 143

Versão mais recente (no momento da confecção deste material): IMAP4 (quarta versão)

Os protocolos POP e IMAP também podem ser chamados de agentes de acesso a mensagens (MAA – Mail Agent Access), ambos utilizam o protocolo de transporte TCP.

Assim como você usa uma chave para as caixas de correio de casa, para acessar a sua caixa no servidor será necessária uma autenticação, por meio de senha pessoal. Após a autenticação, o conteúdo fica disponível para ser lido, apagado ou modificado. Mas as semelhanças entre o POP e o IMAP param por aqui.

## Características do protocolo POP

O POP, mais antigo, tem poucos recursos e se adequava bem às limitações da internet na época em que foi criado. Em linhas gerais, as etapas seguidas são:

**O cliente se conecta ao servidor >> O cliente é autenticado >> Cliente obtém a lista de mensagens armazenadas >> Baixa as mensagens novas >> Apaga ou não as mensagens do servidor >> O cliente encerra a conexão.**

O POP não possui recursos para alteração de mensagens ou para criar e mover entre pastas.

O protocolo POP atende bem a clientes que farão todo o armazenamento do histórico de mensagens no terminal do usuário, fazendo da caixa no servidor apenas um ponto de passagem temporário.



### Atenção

O POP pode não ser adequado para caixas de correio que serão acessadas por vários clientes, pois as mensagens poderiam ser apagadas antes que todos os clientes pudessem baixá-las.

## Mensagens do protocolo POP

O protocolo utiliza comandos de texto para comunicação e é bem simples. Observe o exemplo a seguir:

plain-text

```
(01) S: +OK POP3 server ready
(02) C: USER maria@empresay.com
(03) S: +OK User name accepted, password please
(04) C: PASS senhascretadamaria
(05) S: +OK Mailbox open
(06) C: LIST
(07) S: +OK 2 messages (320 octets)
(08) S: 1 5602
(09) S: 2 1200
(10) S: .
(11) C: RETR 1
(12) S: +OK 5602 octets
(13) S: O SERVIDOR ENVIA O CONTEÚDO DA MENSAGEM 1
(14) S: .
(15) C: DELE 1
(16) S: +OK message 1 deleted
(17) C: RETR 2
(18) S: +OK 1200 octets
(19) S: O SERVIDOR ENVIA O CONTEÚDO DA MENSAGEM 2
(20) S: .
(21) C: DELE 2
(22) S: +OK message 2 deleted
(23) C: QUIT
(24) S: +OK BYE!
```

As linhas iniciadas com “S:” indicam conteúdo enviado pelo servidor (servidor Y). As linhas iniciadas com “C:” indicam o conteúdo enviado pelo cliente (cliente de correio da usuária Maria).

Com facilidade, podemos analisar a conversa pelo protocolo POP e entender o que aconteceu:

- A usuária Maria se conectou ao servidor POP e fez sua autenticação (linhas 1 a 5).
- Em seguida, por meio do comando LIST, foi informada de que duas mensagens estavam presentes em sua caixa de correio (linhas 6 a 10).
- Leu cada uma delas (comando RETR nas linhas 11 e 17) e as apagou em seguida (comando DELE nas linhas 15 e 21).

## Características do protocolo IMAP

O protocolo IMAP foi criado para resolver limitações do POP. Com ele, podemos ter um servidor de correio que armazena as mensagens do usuário indefinidamente e permite organizá-las em pastas. Também possui funções avançadas, como pesquisa de mensagens no servidor (busca por endereços, assunto e texto).

O IMAP permite marcar mensagens (lida, não lida, respondida) e prevê o uso simultâneo de diversas caixas de mensagens para o mesmo usuário, sendo possível mover mensagens entre essas caixas. Os clientes de correio costumam exibir essas diversas caixas como uma árvore de pastas ao usuário, facilitando a organização das mensagens. Quando o volume de mensagens armazenadas é muito grande e definitivo, não apenas temporário, esses são recursos essenciais para o usuário, do contrário seria muito difícil encontrar uma mensagem posteriormente.

## Segurança

Tanto o POP quanto o IMAP podem ser utilizados sem qualquer criptografia. Porém, como já sabemos, essa é uma situação extremamente indesejada, uma vez que o conteúdo e a senha poderão ser capturados. Ambos os protocolos permitem que a comunicação se torne criptografada logo após o estabelecimento da conexão, através do comando **STARTTLS**.

Também é possível que um servidor seja configurado para usar criptografia obrigatória desde o início da conexão. No POP, será usada a porta 995 (POPS) e no IMAP, a porta 993 (IMAPS).

## Webmail

Atualmente, é muito comum acessar o serviço de correio por meio de um navegador web, o que é conhecido como Webmail. Esse acesso ao servidor Webmail é feito por meio do protocolo HTTP (ou HTTPS).

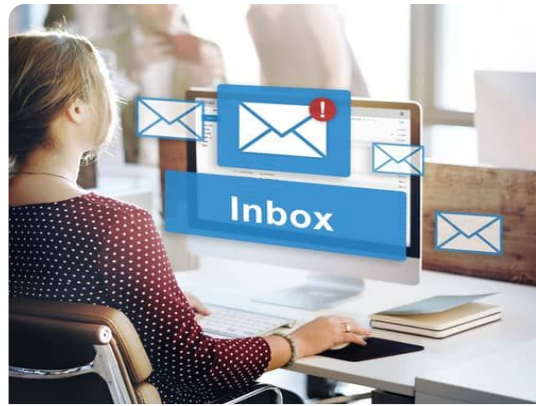


### Atenção

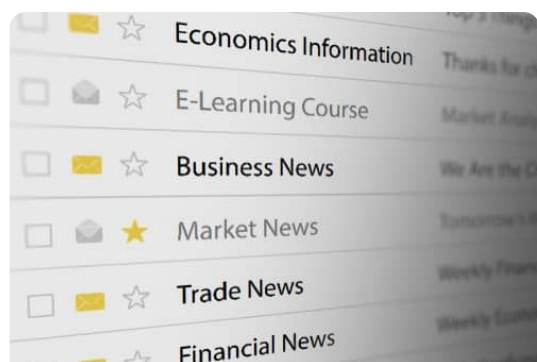
É importante ressaltar que o HTTP não pode ser considerado um protocolo de correio eletrônico.

Quando usamos um Webmail, estamos acessando somente uma página web que se apresenta como um cliente de e-mail. O verdadeiro acesso ao servidor de correio ocorre por trás das cortinas (no chamado backend), invisível para o usuário.

No servidor Webmail, um programa acessa nossa caixa de e-mails no servidor de correio e monta uma página com o conteúdo das mensagens.



Caixa de e-mail



Caixa de e-mail em um Webmail.

# Verificando o aprendizado

## Questão 1

Considere que um servidor de correio eletrônico esteja transmitindo uma mensagem para outro servidor, com um total de dois destinatários naquele domínio. O servidor, no papel de cliente, envia dois comandos RCPT TO:; para um, recebe resposta com código de estado 250 e para o outro, com código 550. Qual deverá ser a ação do servidor?

A

Interromper a transmissão sem enviar a mensagem, preservando a atomicidade das operações do protocolo SMTP, descartando a mensagem e enviando notificação ao remete sobre a impossibilidade do envio.

B

Interromper a transmissão sem enviar a mensagem, preservando a atomicidade das operações do protocolo SMTP, e repetir a tentativa de envio posteriormente.

C

Realizar o envio da mensagem, considerando que um dos destinatários irá recebê-la e enviar mensagem ao remetente, notificando sobre o usuário cujo envio foi recusado pelo servidor de destino.

D

Realizar o envio da mensagem e repetir a ação novamente após alguns minutos, porém só para o destinatário inicialmente recusado.

E

Repetir o envio do comando RCPT TO: até que o servidor responda com código de estado 250, indicando que a caixa do destinatário está disponível.



A alternativa C está correta.

O protocolo SMTP estabelece que respostas com código 5XX para o comando RCPT TO: indicam impossibilidade definitiva para o envio a um destinatário e que o servidor não deverá repetir tal solicitação. Como um dos comandos RCPT TO: teve resposta positiva, a mensagem deverá ser enviada. Será obrigação do cliente (o servidor que está fazendo o envio) notificar o remetente sobre tal erro.

## Questão 2

Escolha a opção que descreve ação realizada por um cliente de correio eletrônico quando o usuário envia uma mensagem.

A

O cliente se conecta ao servidor de correio do destinatário da mensagem utilizando a porta 25.

B

O cliente se conecta ao servidor de correio do destinatário da mensagem utilizando as portas 110 ou 143.

C

O cliente se conecta ao servidor de correio do destinatário da mensagem utilizando a porta 587.

D

O cliente se conecta ao servidor de correio do seu próprio domínio ou provedor utilizando as portas 110 ou 143.

E

O cliente se conecta ao servidor de correio do seu próprio domínio ou provedor utilizando as portas 25 ou 587.



A alternativa E está correta.

O provedor de serviço de correio eletrônico deve possuir um servidor para que seus usuários façam o envio de mensagens, por meio do protocolo SMTP. Atualmente, é aconselhável o uso da porta 587 para esse fim, porém, alguns servidores ainda podem permitir o uso da porta 25. Em todos os casos, é extremamente importante que essas conexões entre cliente de correio e servidor contem com autenticação de usuário e criptografia.

## Protocolo DNS

### Realizando a Tradução de Nomes

No vídeo a seguir, você verá como ocorre o processo de tradução de nomes utilizando o protocolo DNS, a troca de mensagens, os registros utilizados e a associação do DNS com o protocolo HTTP.



#### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Um dos serviços mais importantes da internet é o DNS (Domain Name System), responsável por traduzir nomes amigáveis em números de endereços.

Um dos maiores limitadores para os usuários de qualquer rede é conhecer e referenciar os endereços de estações. Em resumo, se não existir uma forma simplificada de representar os endereços numéricos de cada estação, o uso dessa rede será complexo, desgastante e nada amigável.

Para resolver esse problema, foi desenvolvida uma estrutura para atribuição de nomes de forma hierárquica, permitindo que instituições e empresas tenham o seu domínio de nomes e controle sobre as atribuições dentro desse domínio.

Por exemplo, a Empresa X, adquire o domínio “empresax.com” e cria os seus nomes sob esse domínio.

- www.empresax.com
- portal.empresax.com
- webmail.engenharia.empresax.com

Cada componente de um nome é separado por um ponto. No terceiro exemplo, vemos que o domínio pode ser ainda subdividido hierarquicamente entre diferentes unidades ou departamentos.

Essa estrutura, chamada de **Name Space**, é estabelecida como uma árvore e cada um dos nós (inclusive as folhas) pode possuir registros (Resource Records).

O componente mais à direita é o domínio de topo (TLD – Top Level Domain). O TLD pode ser:



### TLDs genéricos

com (para fins comerciais), edu (instituições de ensino), org (organizações sem fins lucrativos).

### TLDs de países

br (Brasil), ar (Argentina), it (Itália), au (Austrália) – cada país tem sua entidade para definir as regras e gerenciar o TLD.

A atribuição de nomes vem dos primórdios da internet, mas com essa solução veio um novo problema: como divulgar esses nomes?

A primeira solução foi a criação de uma grande “lista telefônica”, divulgada diariamente e que precisava ser baixada diretamente em todos os hosts conectados à rede. Geralmente, como os enlaces eram muito lentos, em cada instituição uma pessoa se encarregava de baixar e disponibilizar localmente para os demais usuários. Obviamente, percebemos dois grandes problemas com esse modelo: o trabalho imposto a todos os usuários da rede; e o tempo entre a criação de um nome e o seu “aprendizado” pelos demais usuários.

Para resolver esse problema, foi desenvolvido o serviço de DNS, para que as consultas por nomes pudessem ser feitas em tempo real, quando fossem necessárias. E, mais importante, de forma distribuída, sem um banco de dados central.

O elemento responsável por receber e responder consultas por nomes é o **SERVIDOR DNS**.

Quando uma aplicação cliente (por exemplo, um navegador web) precisa traduzir o nome “webmail.empresax.com” em seu endereço IP, primeiro consultará o servidor DNS. Só depois que receber a resposta saberá o endereço que procura e poderá enviar um primeiro pacote para se conectar a ele.

### Tipos de consulta DNS

Uma das principais configurações de rede de um sistema operacional são os endereços dos servidores de DNS para onde as consultas serão encaminhadas. Geralmente, utilizam-se servidores na rede do próprio provedor ou empresa, pois as respostas tendem a ser mais rápidas.



Consulta a servidor DNS Local.

Esse tipo de consulta é chamado de **recursiva**, pois o cliente pede ao servidor para resolver o nome procurado. Mas para descobrir o endereço procurado, o servidor DNS usará um tipo de consulta chamado de **iterativo**.

Como já vimos, as informações de DNS estão distribuídas, então, a busca envolverá não apenas uma, mas algumas consultas. Funciona assim:

1

Procuramos o endereço: `webmail.engenharia.empresax.com`.

2

O servidor DNS que tem a resposta para a nossa pergunta é aquele que tem a **autoridade** sobre o domínio `engenharia.empresax.com`. Porém, primeiro precisamos descobrir o seu endereço.

3

O servidor de DNS do nosso provedor precisará de um ponto de partida. Ele começará a pesquisa, fazendo a pergunta para um **SERVIDOR RAIZ** (root server). Ao todo, existem 13 servidores raiz, cada um com diversas réplicas espalhadas pelo mundo.

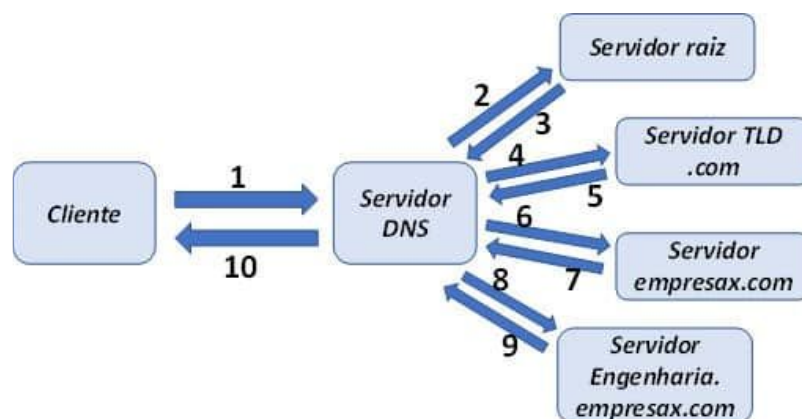
4

O servidor raiz não sabe a resposta para a consulta. Ele não sabe o endereço do nome procurado. Mas ele sabe o endereço do servidor DNS com autoridade sobre o TLD, no caso o “.com”.

5

O nosso servidor DNS, então, entrará em contato com o servidor do TLD, que também não sabe a resposta para a consulta. Mas sabe o endereço do servidor DNS com autoridade sobre o domínio “`empresax.com`”. Avançando um componente por vez, nosso servidor DNS resolverá o nome que procuramos e só então nos responderá.

A imagem a seguir ilustra o processo descrito.



Pesquisa por um nome pelo método iterativo.

Em teoria, seria possível fazer toda essa consulta pelo método recursivo, porém, não é utilizado na prática, inclusive por questões de segurança.

O protocolo DNS pode utilizar as camadas de transporte TCP ou UDP, e o servidor sempre responde na porta 53. A maioria das consultas é realizada com o protocolo UDP, que não exige estabelecimento de conexão, tornando a interação mais ágil. O tamanho máximo da resposta é de 512 bytes (descontando os cabeçalhos UDP e IP). O protocolo TCP é usado quando a resposta ultrapassa o limite de 512 bytes, sendo truncada.

## Os registros DNS (RR – Resource Registry)

Neste vídeo, mergulhe no mundo dos registros DNS (Resource Registry) e descubra como eles são essenciais para a internet funcionar. Explore os principais tipos de registros, como A, AAAA, CNAME, MX e mais. Aprenda como os servidores de DNS gerenciam zonas e mantêm caches atualizados para uma navegação eficiente.



### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

O serviço de DNS permite que cada nó da árvore possua zero ou mais registros de recursos. Os principais tipos de registro estão descritos na tabela a seguir:

Registro de Recursos (RR)	Descrição
A	Representa um número de endereço IPv4 (32 bits)
AAAA	Representa um número de endereço IPv6 (128 bits)
CNAME	Nome canônico, permite mapear para outro nome. Esse tipo de registro pode ser usado para facilitar a administração de nomes e endereços de servidores.  Por exemplo, o nome “webmail.engenharia.empresax.com” pode ter um CNAME para “server1821.provedorxpto.com”, onde o serviço está hospedado. Assim, o administrador da zona “empresax.com” não precisa se preocupar em saber o endereço IP do servidor, nem modificar o registro se o “provedorxpto.com” mudar o endereço desse servidor.
MX	Mail Exchanger. Indica o endereço do servidor para onde e-mails para determinado domínio devem ser encaminhados.
NS	Name Server. Indica os servidores com autoridade sobre o nome ou domínio consultado.
SOA	Start of Authority. Registro que contém diversas informações administrativas sobre uma zona. Entre elas, configurações para a transferência de zonas entre servidores secundários.
TXT	Permite incluir quaisquer informações, para fins diversos. Costuma ser usado para fins de segurança no envio de e-mails (SPF e DKIM), verificar a titularidade sobre um nome, por exemplo.
PTR	Usado para consultas em zona reversa, permitindo consultar o nome associado a um endereço.

Tabela: Principais tipos de registro DNS.  
Elaborado por Fernando Diniz Hämmerli.

Um servidor de DNS trabalha basicamente com dois conjuntos de dados:

## Zonas

São as bases de dados com todos os domínios para os quais o servidor tem autoridade.

## Cache

É o conjunto de registos consultados pelo servidor recentemente, tornando desnecessário repetir consultas. Esses registos são descartados depois de algum tempo, evitando que o cache tenha informações desatualizadas. Esse tempo é definido para cada registo ou domínio, sendo conhecido como TTL (Time to Live).

## Zonas direta e reversa

O serviço de nomes trabalha com dois tipos de zona:

### Direta

A zona direta é a que temos usado até agora como exemplo, em que a árvore representa os domínios e nomes, e os registos podem representar endereços.



### Reversa

A zona reversa trabalha ao contrário, permitindo pesquisar por um endereço e obter um nome associado.

Em geral, essas zonas são configuradas por quem tem autoridade sobre um bloco de endereços IP. Para manter a lógica de organização hierárquica, os nomes da pesquisa são formados pelos octetos de um endereço IPv4, na ordem inversa, e com domínio in-addr.arpa.



### Exemplo

30.1.27.183.in-addr.arpa → Representando o endereço IPv4 '183.27.1.30'

O Registro PTR associado indicará um nome para o endereço IP. Desse modo, o detentor do bloco 183.27.1/24 pode administrar a zona reversa para o bloco de endereços, criando registos para cada endereço.

Para endereços IPv6, a lógica é a mesma, porém, cada componente do nome é formado por 4 bits (um caractere em representação hexadecimal, de 0 a f), e o domínio é ip6.arpa.



### Exemplo

6.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.4.0.0.0.0.0.f.f.2.1.1.0.0.2.ip6.arpa → Representando o endereço IPv6 '2001:12ff:0:4::6'

# Protocolo DHCP

## Descomplicando o Protocolo DHCP

Neste vídeo, explore o mundo do Protocolo DHCP (Dynamic Host Configuration Protocol). Saiba como esse protocolo revolucionou a forma como os dispositivos se conectam à rede, eliminando a necessidade de configuração manual. Descubra como funciona a negociação entre cliente e servidor DHCP e entenda o conceito de "lease time" para o gerenciamento eficiente de endereços IP em redes de todos os tamanhos.



### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Nos primórdios da internet, cada vez que um dispositivo precisava se conectar a uma rede, o seu administrador era responsável por definir um endereço para ele e configurá-lo manualmente. Esse modo de trabalho tem alguns inconvenientes, como:

- Necessidade de intervenção manual no dispositivo.
- Controle dos endereços alocados a cada dispositivo.

Mas a pior consequência é a necessidade de alocar um endereço para cada dispositivo que possivelmente se conectará à rede e não aos que estejam efetivamente conectados em dado momento.

Para resolver esses problemas, tornando a conexão do dispositivo algo simples, foi desenvolvido o protocolo DHCP (Dynamic Host Configuration Protocol), ou Protocolo de configuração dinâmica de estações, em tradução livre.

Em resumo, o DHCP é um protocolo que permite a um servidor na rede configurar automaticamente todas as estações que a ele se conectarem.



### Dica

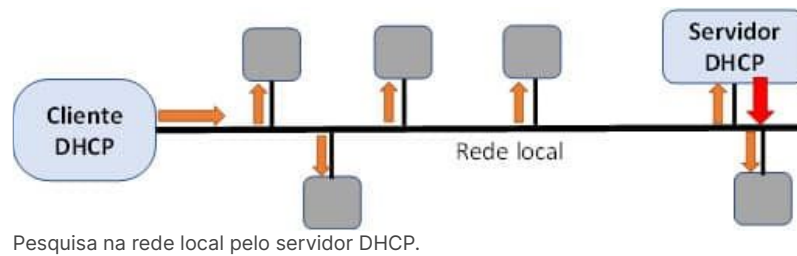
A configuração manual continua sendo possível, mas só é feita em dispositivos permanentes, em que o endereço não vai ou não pode mudar. Por exemplo, em servidores.

## Funcionamento do DHCP

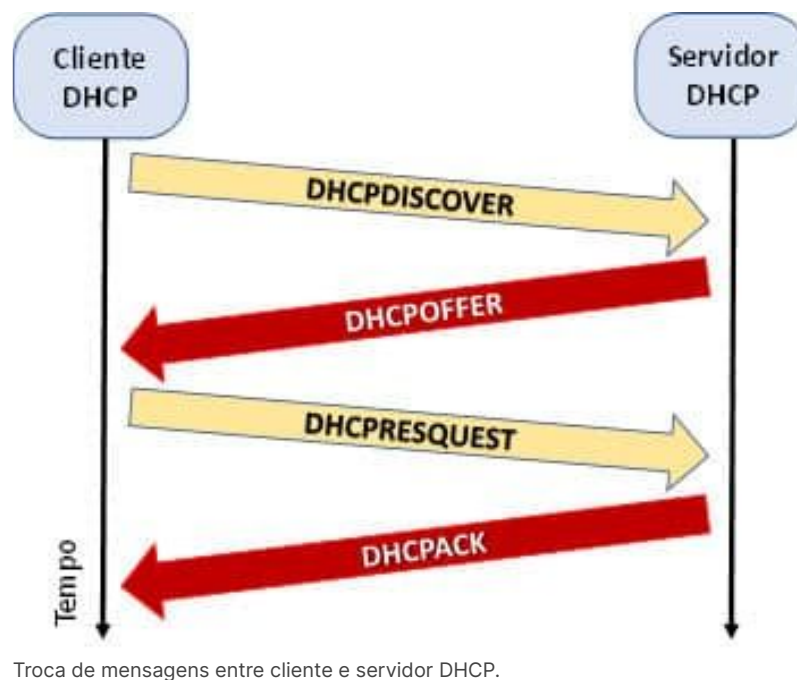
O DHCP é baseado na figura do servidor DHCP, elemento central que controla e distribui endereços aos dispositivos conectados.

O algoritmo do protocolo DHCP permite que um cliente se conecte na rede sem qualquer configuração prévia de rede ou servidor. Na prática, a única configuração necessária no dispositivo é para que ele tente obter seu endereço por meio do DHCP, tipicamente uma configuração de fábrica dos principais sistemas operacionais no mercado.

Quando um dispositivo se conecta a uma rede, ele inicia o processo de **descoberta** do protocolo DHCP. Esse processo é feito com um pacote transmitido em **broadcast** na rede, ou seja, recebido por todas as demais estações conectadas na rede. Apenas o servidor DHCP deve responder a essa solicitação.



Na resposta, o servidor DHCP se identifica e oferece um endereço para uso dessa estação. A negociação continua com o cliente requisitando o uso do endereço IP oferecido e o servidor responde com as informações necessárias para a configuração da rede.



Veja a seguir, os parâmetros dos pacotes DHCP.

Pacote	Emissor	Endereço de origem	Endereço de destino	Protocolo de transporte	Porta de origem	Porta de destino
DHCPDISCOVER	Cliente	0.0.0.0	255.255.255.255	UDP	68	67
DHCPOFFER	Servidor	IP do servidor DHCP	IP oferecido ao cliente	UDP	67	68
DHCPREQUEST	Cliente	0.0.0.0	255.255.255.255	UDP	68	67
DHCPACK	Servidor	IP do servidor DHCP	IP concedido ao cliente	UDP	67	68

O servidor DHCP informa ao cliente o endereço IP que deve ser usado, máscara de rede, endereço de gateway e servidores de DNS. Também pode ser usado para passar configurações adicionais como servidor para ajustar o relógio.

## Gerenciamento de endereços

O DHCP foi projetado para funcionar em redes pequenas e grandes, em que há grande volume de dispositivos chegando e saindo. Um dos problemas desse cenário é o esgotamento dos endereços de rede, visto que um endereço é alocado para cada dispositivo. A solução adotada pelo DHCP foi criar um prazo de vida para cada concessão. O dispositivo conectado na rede, continua usando o mesmo endereço, porém, o dispositivo que se desconecta perde a concessão após algum tempo e quando fizer uma nova requisição ao servidor DHCP poderá receber outro endereço.

Esse prazo de vida é chamado de **lease time**, que é definido pelo administrador do servidor DHCP e informado ao cliente no ato da concessão. O cliente deve solicitar a renovação da concessão do endereço após a metade do lease time.



### Exemplo

Se o lease time for de 60 minutos, o cliente deverá solicitar o uso do endereço novamente ao servidor DHCP após 30 minutos.

Após o lease time esgotar, o servidor considera o endereço como livre e pode ser concedido a outros clientes.

O dimensionamento do lease time é uma questão estratégica, que deve levar em consideração a quantidade de endereços disponíveis (o tamanho da rede) e a rotatividade de dispositivos na rede.

#### Lease time longo

A rede de uma empresa em que os dispositivos são quase sempre os mesmos e eventualmente algum é substituído, pode ter um lease time longo, como de um dia ou uma semana.



#### Lease time curto

Uma rede de convidados em local movimentado, como em um aeroporto, vai precisar de tempos mais curtos, da ordem de minutos, para evitar que em algum momento todos os endereços estejam alocados para dispositivos, inclusive que já tenham se desconectado da rede.



### Atenção

O lease time é sempre tratado na unidade de segundos pelo protocolo DHCP e não é aconselhável que seja extremamente baixo, pois aumentará em muito o tráfego de requisições ao servidor.

## Protocolo SNMP

### SNMP: Monitorando Redes para Garantir um Bom Serviço

Neste vídeo, mergulhe no mundo do Protocolo SNMP (Simple Network Management Protocol). Descubra como as equipes de gerenciamento de redes usam o SNMP para monitorar equipamentos e serviços, prevenir falhas e manter uma rede estável. Explore a arquitetura SNMP, seus elementos-chave e os modos de operação, incluindo consultas e notificações. Saiba como o SNMP ajuda a manter sua conexão de internet suave e sem interrupções.



#### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Quando estamos em casa navegando na internet ou assistindo a um vídeo pelo streaming, não nos preocupamos com tudo o que está acontecendo na rede da operadora ou do provedor para que o nosso acesso continue funcionando. Não vemos que por trás há uma grande equipe operacional dedicada para garantir um bom serviço ao assinante.



Falha na conexão.

Um dos pontos mais críticos de uma rede é o seu **gerenciamento**, que inclui a monitoração dos equipamentos e serviços. Uma boa gerência é aquela que consegue evitar ao máximo todos os impactos ao funcionamento da rede. Ela deve ser capaz de detectar falhas inesperadas, grandes ou pequenas, prever e corrigir problemas antes de causarem algum transtorno.



### Exemplo

A falha no ventilador de um equipamento pode ser detectada precocemente e corrigida antes de esse equipamento deixar de funcionar. Enlaces adicionais podem ser ativados para aliviar o tráfego em outros enlaces, antes que fiquem totalmente congestionados.

Para esse monitoramento, as equipes contam com ferramentas de software sofisticadas, capazes de agregar e processar dados de inúmeros equipamentos e serviços, gerar gráficos e dados para acompanhamento e disparar alertas em casos em que uma ação imediata é necessária.



## Arquitetura SNMP e seus elementos

Vamos conhecer os principais elementos do protocolo SNMP.

### Estação de gerenciamento de rede

Executam ferramentas de gerenciamento que monitoram e controlam outros equipamentos.

### Elementos de redes

Equipamentos, como switches, roteadores, servidores, impressoras, estações de trabalho, gerenciados por meio do agente de gerenciamento.

### Agente de gerenciamento

Programas executados nos elementos de rede e que trocam informações com as estações de gerenciamento de rede. Os agentes podem enviar dados sobre o equipamento e alertas em condições especiais.

### Protocolo de gerenciamento de rede

Por meio dele, agentes e estações de gerenciamento trocam informações. SNMP (Simple Network Management Protocol), ou protocolo simples de gerenciamento de rede, em tradução livre.

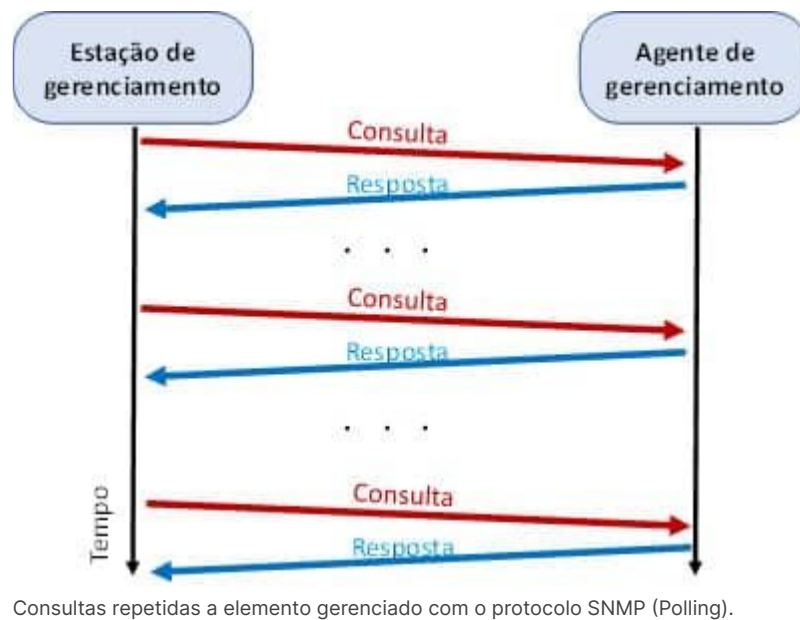
### Base de informações de gerenciamento

Os recursos a serem gerenciados são representados como objetos, e a coleção objetos é referenciada como Base de Informações Gerenciais (MIB).

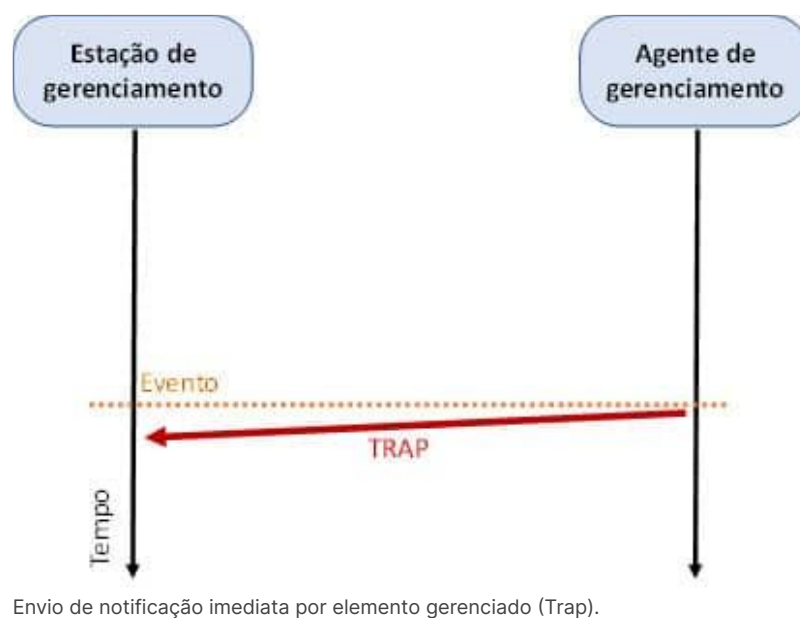
## Modos de operação do SNMP

A estação de gerenciamento de rede pode buscar as informações sobre um elemento gerenciado por meio das consultas, com o protocolo SNMP. A cada pergunta, a estação coleta dados para o monitoramento sobre um ou mais recursos gerenciados. Por exemplo, pode fazer consultas sobre o uso instantâneo de uma interface de rede, consumo de memória ou temperatura do processador.

Em geral, as consultas são feitas periodicamente pela estação de gerenciamento, um modelo conhecido como **polling**.



O agente de um elemento também pode ser configurado para notificar uma estação de gerenciamento imediatamente após a ocorrência de algum evento. Esse tipo de notificação é chamado de **trap**. Pode ser usado, por exemplo, para notificar uma falha em um componente eletrônico do elemento ou uma tentativa de acesso indevido.



A estratégia de gerenciamento pode combinar os dois mecanismos, aproveitando o melhor de cada um.

Veja as vantagens e desvantagens do polling a seguir.

#### Vantagem

A estação de gerenciamento mantém contato periódico com o elemento monitorado, portanto, será capaz de perceber se ela ficar indisponível, parando de responder às consultas.



#### Desvantagem

Gera um tráfego constante de monitoramento, que cresce com a quantidade de elementos monitorados, e o intervalo entre as consultas. A estação de gerenciamento também pode demorar a saber de um evento ocorrido entre duas consultas.

O trap permite notificar um evento com mais rapidez, porém, como o SNMP utiliza o protocolo de transporte UDP, não há como garantir que o pacote com a notificação foi recebido.

Os agentes escutam a porta 161 para as consultas e enviam traps destinados à porta 162.

## Verificando o aprendizado

### Questão 1

**Quando um servidor DNS consulta um servidor raiz, ele espera receber como resposta:**

A

O registro (RR) do nome sendo pesquisado.

B

O endereço do servidor DNS com autoridade sobre o domínio de topo (TLD).

C

O endereço do servidor DNS com autoridade sobre o nome sendo pesquisado.

D

O registro (RR) armazenado em cache do nome sendo pesquisado.

E

O TTL de um registro (RR), definindo o tempo pelo qual esse registro poderá ser mantido em cache.



A alternativa B está correta.

Os servidores raiz (root servers) são o ponto de partida para a resolução de nomes. Por meio deles, obtemos a indicação dos servidores (NS) com autoridade sobre o domínio de topo (TLD) do nome pesquisado.

### Questão 2

Considerando o que aprendemos sobre o processo de renovação de concessão do DHCP, responda: em uma rede em que a concessão de endereços possui um lease time de 20 minutos, quantas renovações uma estação conectada continuamente fará ao longo de uma hora?

A

2

B

3

C

4

D

5

E

6



A alternativa E está correta.

O valor padrão que o DHCP estabelece para a solicitação de renovação é metade do lease time. Sendo assim, se o lease time é de 20 minutos, o cliente fará uma solicitação de renovação a cada 10 minutos. Portanto, serão seis renovações ao longo de uma hora.

## Sniffer

### Realizando a captura e análise de pacotes

No vídeo a seguir, você verá o Wireshark e como ocorre o processo para realizar a captura de tráfego e sua respectiva análise.



#### Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Para compreender bem o funcionamento dos protocolos utilizados nas redes de computadores, é importante ter o conhecimento de como as informações são trocadas entre hospedeiros e quais são essas informações. De modo a solidificar o conhecimento teórico que você adquiriu nos módulos anteriores, utilizaremos um analisador de pacotes de rede que nos permitirá capturar e analisar as informações que passam pela interface de rede do computador.

A ferramenta usada para observar os pacotes trocados entre os hospedeiros é conhecida como **sniffer** de pacotes. Um sniffer captura os pacotes que passam pela interface de rede do computador para armazenar e exibir o conteúdo dos campos dos protocolos que compõem os pacotes capturados.



Estrutura de um sniffer.

No lado direito da imagem estão os protocolos (no caso, protocolos da internet) e aplicativos que normalmente são executados no hospedeiro. O sniffer, mostrado dentro do retângulo tracejado, é uma adição ao software do hospedeiro e consiste em duas partes.

#### Biblioteca

A biblioteca de captura de pacotes recebe uma cópia de cada quadro da camada de enlace que é enviado ou recebido pelo hospedeiro. Ela contém todas as mensagens enviadas e recebidas por todos os protocolos e aplicativos em execução no hospedeiro.

#### Analisador

O analisador de pacotes exibe os encapsulamentos e o conteúdo de todos os campos dos protocolos presentes. Para fazer isso, o analisador de pacotes deve conhecer a estrutura de todas as mensagens trocadas pelos protocolos.

Para a análise dos protocolos, utilizaremos o **Wireshark**, que é um analisador de protocolos gratuito executado em computadores Windows, Linux/Unix e Mac. Com ele, podemos analisar todos os protocolos que passam por uma interface de rede, assim como os encapsulamentos que ocorrem para a transmissão de dados.

Neste módulo, estão propostas sete atividades de análise de protocolos de aplicação estudados no conteúdo, que usarão o Wireshark ou a ferramenta de desenvolvedor do navegador.

## HTTP – com Wireshark

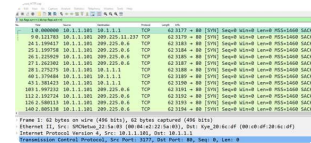
Nesta atividade, usaremos o Wireshark para estudar as conexões e os pacotes durante uma troca de mensagens HTTP, com a transferência de arquivos.

Para isso, baixe o arquivo com a [captura de pacotes](#) e abra com o Wireshark ou ative a captura no Wireshark e acesse um site no seu navegador.

Agora, analisaremos algumas informações da troca de mensagens realizada na captura que foi disponibilizada.

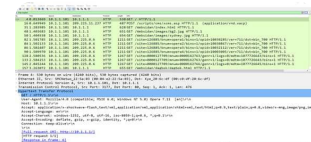
### Conexões TCP

Utilize o filtro `tcp.flags.syn==1 && tcp.flags.ack==0` para que o Wireshark mostre somente o primeiro pacote de cada conexão TCP (Flag SYN=1 e ACK=0). Ao todo, quantas conexões TCP foram feitas entre o cliente e o servidor? Perceba que existem várias aberturas de conexão para diversos servidores, todos direcionados à porta 80, portanto, para o servidor web.



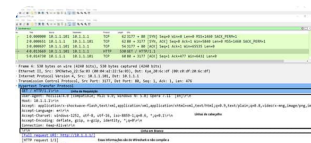
### Requisições

Quantas requisições foram feitas ao servidor? Você pode utilizar o filtro `http.request` para que sejam exibidas apenas as requisições HTTP. Verifique que foram enviadas mensagens de requisição GET e POST.



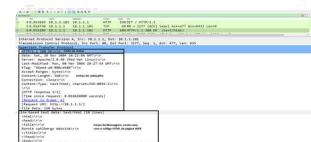
### Requisição e resposta

Observe as mensagens de requisição e resposta, percebendo seu formato, com as linhas de requisição e resposta, linhas de cabeçalho e corpo da mensagem.



### Conteúdo das mensagens

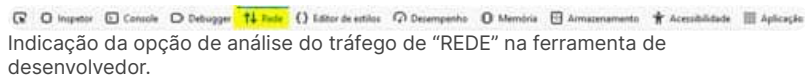
Agora, imagine quais seriam as consequências para esta atividade se o site utilizasse o protocolo HTTPS? Você conseguiria visualizar o conteúdo das mensagens?



## HTTP – com ferramenta de desenvolvedor

Os principais navegadores possuem uma ferramenta para desenvolvedores, que permite analisar em detalhes todos os aspectos de um site, desde as conexões na rede até sua composição visual.

Essa ferramenta é ideal para analisar conexões de HTTPS, uma vez que ela nos mostrará as requisições, respostas, os cabeçalhos e conteúdo sem criptografia, permitindo a sua análise. Abra o seu navegador (preferencialmente Google Chrome, Mozilla Firefox ou Microsoft Edge) e pressione a tecla F12 (dependendo do navegador pode variar). Selecione a aba “Rede” (Network).

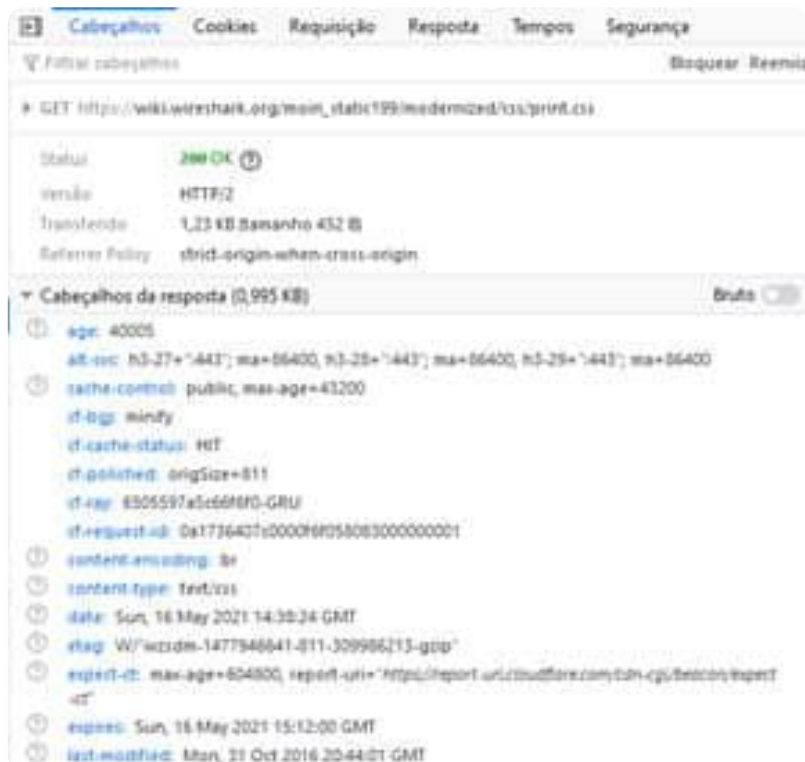


Após abrir a aba, acesse um site, de sua preferência. Observe as linhas se preenchendo, uma para cada requisição.

Status	Método	Domínio	Argumento
200	GET	wiki.wireshark.org	SampleCaptures
200	GET	fonts.googleapis.com	css?family=Open+Sans
200	GET	fonts.googleapis.com	css?family=Open+Sans+Condensed:300,700
200	GET	wiki.wireshark.org	common.js
200	GET	wiki.wireshark.org	screen.css
200	GET	wiki.wireshark.org	ws-wiki-screen.css
200	GET	wiki.wireshark.org	print.css
200	GET	wiki.wireshark.org	projecthon.css
200	GET	wiki.wireshark.org	ws-wiki-print.css
200	GET	fonts.gstatic.com	woff2/Cs126A2y6A-UPV23b-woff2
200	GET	www.wireshark.org	wirebadge@10x57.png
200	GET	wiki.wireshark.org	wire4.png
200	GET	wiki.wireshark.org	wire5.png
200	GET	wiki.wireshark.org	wire6.png
200	GET	fonts.gstatic.com	woff2/Cs126A2y6A-UPV23b-woff2
200	GET	wiki.wireshark.org	favicon.ico

Exemplo do conteúdo exibido pela ferramenta do desenvolvedor após captura de tráfego.

Selecione uma requisição e observe na janela ao lado os detalhes das requisições e respectivas respostas. Alguns navegadores têm a opção **Bruto**, que permite visualizar o cabeçalho no formato original.



Analise os dados exibidos pela ferramenta e procure responder as seguintes questões:

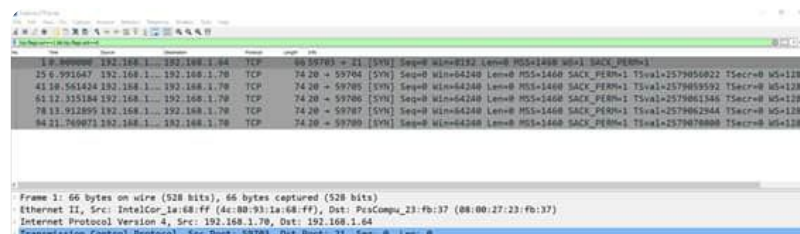
- Quantas requisições, ao todo, foram realizadas?
- Quais foram os códigos de estado para as requisições? Alguma apresentou código de erro? Por quê?
- Quais tipos de recursos foram obtidos pelo navegador?
- Acesse um site que disponibilize um campo para preenchimento. Pode ser um campo de pesquisa ou login e senha. Analise os dados obtidos pela ferramenta ao enviar. Busque entre as requisições, aquela com método POST.

## FTP

Nesta atividade, usaremos o Wireshark para estudar as conexões e os pacotes durante uma sessão FTP, com a transferência de arquivos. Baixe o arquivo com a [captura de pacotes](#).

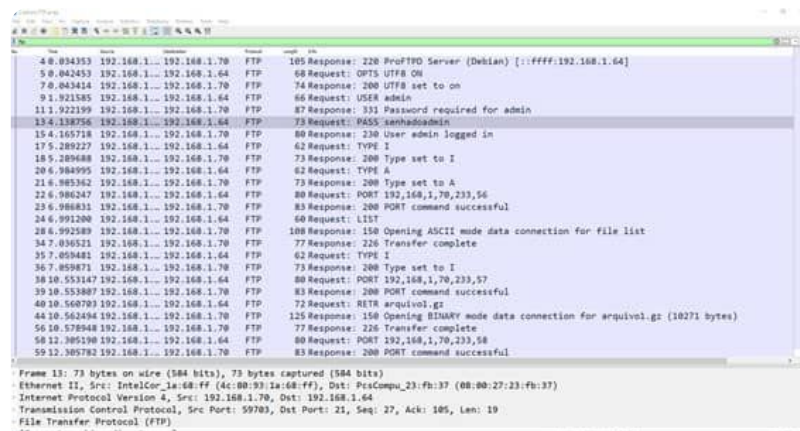
Agora, analisaremos algumas informações da troca de mensagens realizada na captura que foi disponibilizada.

Novamente, utilize o filtro “tcp.flags.syn==1 && tcp.flags.ack==0” para que o Wireshark mostre somente o primeiro pacote de cada conexão TCP (Flag SYN=1 e ACK=0). Ao todo, quantas conexões TCP foram feitas entre o cliente e o servidor?



Captura de tela do software Wireshark

Filtre agora pelo protocolo FTP. Qual o objetivo da segunda conexão TCP? Que dados foram transferidos?



Captura de tela do software Wireshark



Agora, siga as etapas adiante:

1. Verifique o arquivo e veja quais arquivos foram transferidos. Quantos foram upload e quantos foram download?
2. Verifique se o protocolo estava operando no modo ativo ou passivo.
3. Considerando que o FTP não utiliza criptografia, você consegue identificar o login e a senha usados para acesso ao servidor?

## TELNET

Nesta atividade, estudaremos um arquivo de captura com um acesso ao serviço TELNET.

Cliente

**192.168.1.70**

Servidor

**192.168.1.64**

A tela de toda a sessão foi gravada e está copiada abaixo. Todos os textos foram enviados do servidor para o cliente, com exceção dos textos **"admin"**, **"ls /tmp"**, **"cp /etc/lsb-release ."**, **"cat lsb-release"** e **"exit"** que foram enviados do cliente para o servidor.

plain-text

```
*****
* Servidor de Testes *
* *
* Acesso restrito a administradores *
*****
```

```
servidor login: admin
Password:
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.4.0-31-generic x86_64)
```

```
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
```

```
* MicroK8s passes 9 million downloads. Thank you to all our contributors!
```

```
https://microk8s.io/
```

```
11 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable
```

```
Last login: Sat May 15 16:21:06 UTC 2021 from 192.168.1.70 on pts/0
To run a command as administrator (user "root"), use "sudo ".
See "man sudo_root" for details.
```

```
admin@servidor:~$ ls /tmp
snap.lxd
systemd-private-4dd24f00a8634850966e9e9bf3e20647-fwupd.service-cso5Lh
systemd-private-4dd24f00a8634850966e9e9bf3e20647-systemd-logind.service-gcwmKh
systemd-private-4dd24f00a8634850966e9e9bf3e20647-systemd-resolved.service-XLdZxi
systemd-private-4dd24f00a8634850966e9e9bf3e20647-systemd-timesyncd.service-USIYZi
admin@servidor:~$ cp /etc/lsb-release .
admin@servidor:~$ cat lsb-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=20.04
DISTRIB_CODENAME=focal
DISTRIB_DESCRIPTION="Ubuntu 20.04 LTS"
admin@servidor:~$ exit
```

Baixe o arquivo com a [captura de pacotes](#) e para reconstruir toda a comunicação pelo protocolo TELNET, clique sobre um dos pacotes com o botão direito e selecione "Follow → 'TCP Stream'". Repare que o protocolo não usa qualquer tipo de criptografia e todo o conteúdo da comunicação está visível.

Agora, analisaremos algumas informações da troca de mensagens realizada na captura que foi disponibilizada.

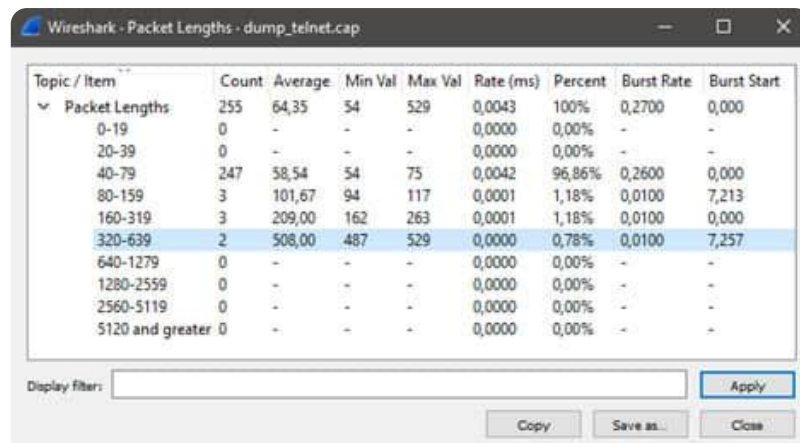
Utilizando as ferramentas de estatísticas do Wireshark, verifique a quantidade de pacotes por faixa de tamanho. No menu, clique em "Statistics" → "Packet Lengths".

Será gerada a tela a seguir. Na primeira coluna, são indicadas faixas de tamanho dos pacotes; e na segunda coluna, a contagem de pacotes em cada faixa.

Por que a maioria dos pacotes (cerca de 96%) tem tamanho reduzido, entre 40 e 79 bytes? Qual característica do protocolo leva a esse comportamento?

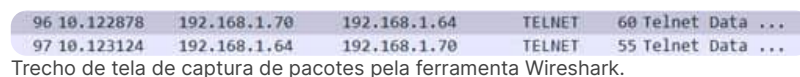
Ao analisar a captura é possível perceber muitos pacotes em pares, sendo um em cada sentido, e de tamanho reduzido, às vezes, com apenas 1 byte transportado.

Qual a causa desse efeito? Qual o teor desse conteúdo?



Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
Packet Lengths	255	64,35	54	529	0,0043	100%	0,2700	0,000
0-19	0	-	-	-	0,0000	0,00%	-	-
20-39	0	-	-	-	0,0000	0,00%	-	-
40-79	247	58,54	54	75	0,0042	96,86%	0,2600	0,000
80-159	3	101,67	94	117	0,0001	1,18%	0,0100	7,213
160-319	3	209,00	162	263	0,0001	1,18%	0,0100	0,000
320-639	2	508,00	487	529	0,0000	0,78%	0,0100	7,257
640-1279	0	-	-	-	0,0000	0,00%	-	-
1280-2559	0	-	-	-	0,0000	0,00%	-	-
2560-5119	0	-	-	-	0,0000	0,00%	-	-
5120 and greater	0	-	-	-	0,0000	0,00%	-	-

Tela de análise de pacotes pela ferramenta Wireshark, com contagem por faixa de tamanho.



96	10.122878	192.168.1.70	192.168.1.64	TELNET	60 Telnet Data ...
97	10.123124	192.168.1.64	192.168.1.70	TELNET	55 Telnet Data ...

Trecho de tela de captura de pacotes pela ferramenta Wireshark.

## SMTP

Nesta atividade, usaremos o Wireshark para estudar as conexões e os pacotes durante uma sessão SMTP. Baixe o arquivo com a [captura de pacotes](#).

Agora, analisaremos algumas informações da troca de mensagens realizada na captura que foi disponibilizada. Digite SMTP no filtro para analisar as questões.

1. A conexão SMTP presente foi de um cliente de correio para um servidor ou entre dois servidores de correio eletrônico? Como é possível afirmar isso?
2. Analise as informações e identifique qual é o remetente da mensagem.
3. Você consegue identificar qual é o assunto da mensagem?

## DNS

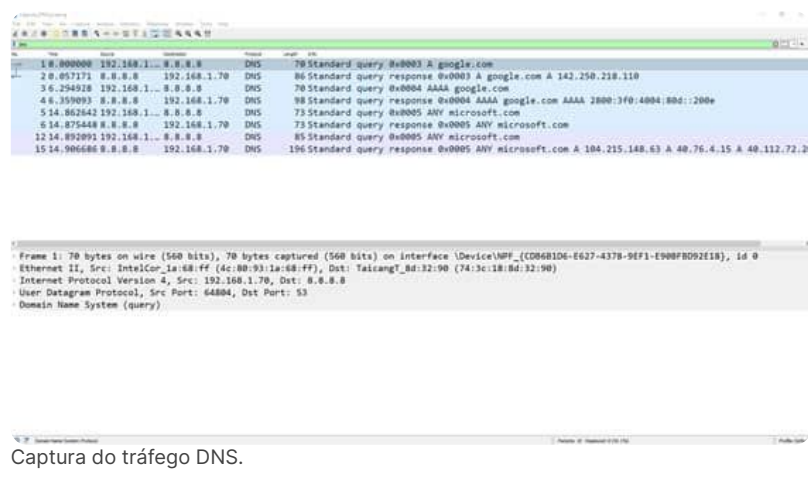
Nesta atividade, estudaremos o protocolo DNS usando duas ferramentas. A primeira é o nslookup. Trata-se de um cliente DNS, já instalado nos principais sistemas operacionais, que nos permite fazer consultas diretamente a servidores DNS e observar sua resposta. Em paralelo, usaremos o Wireshark para capturar o tráfego gerado pelo nslookup e depois analisar seus pacotes.

Para fazer essa atividade, inicie a captura com o programa Wireshark. Execute o programa nslookup. No Windows digite “nslookup” no menu iniciar ou no Prompt de comandos. No nslookup, digite os comandos a seguir e observe as respostas:

- server 8.8.8.8 → O servidor DNS para onde as consultas serão enviadas.
- set type=a → Pesquisar por registros de tipo A (endereço IPv4).
- google.com → Pesquisar esse nome.
- set type=aaaa → Pesquisar por registros de tipo AAAA (endereço IPv6).
- google.com → Pesquisar esse nome.
- set type=any → Pesquisar por todos os tipos de registro.
- microsoft.com → Pesquisar esse nome.
- exit → Sair do programa.

Interrompa a captura do Wireshark e filtre pelo protocolo DNS.

Se preferir, você pode baixar a [captura de pacotes](#) utilizada na atividade. A captura será semelhante a imagem a seguir.



Captura do tráfego DNS.

Agora, analisaremos algumas informações da troca de mensagens realizada na captura que foi disponibilizada.

- Analise as informações e identifique quais os endereços do nome ‘google.com’ (IPv4 e IPv6).
- Observe os pacotes capturados e relacione as consultas e respectivas respostas.
- Encontre nos pacotes de consulta (queries) as informações solicitadas ao servidor.
- Encontre nos pacotes de resposta (answers) as informações exibidas pelo nslookup.
- As consultas foram realizadas com protocolo de transporte UDP ou TCP? Se o TCP foi usado, identifique o motivo.

# DHCP

Nesta atividade, analisaremos os pacotes trocados entre cliente e servidor DHCP para obter uma concessão (lease) de endereço.

Para realizar a atividade, inicie a captura com o programa Wireshark. Utilizando o Windows, abra o Prompt de comandos e digite os seguintes comandos:

- `ipconfig /release` → Liberar ("esquecer") a atual concessão DHCP.
- `ipconfig /renew` → Obter uma nova concessão.
- `ipconfig` → Exibir a configuração de rede obtida.

Agora, interrompa a captura com o programa Wireshark e filtre os pacotes, utilizando a expressão `dhcp`.

Observe que o primeiro pacote capturado foi o DHCPRELEASE (quando foi digitado o primeiro comando). Ignore esse pacote nesta atividade.

Se preferir, você pode baixar a [captura de pacotes](#).

Agora, analisaremos algumas informações da troca de mensagens realizada na captura que foi disponibilizada.

- Identifique quantos pacotes foram trocados entre cliente e servidor DHCP.
- Para cada pacote, observe os endereços e as portas de origem e destino e compare com a tabela apresentada no módulo 3.
- Encontre nos pacotes de resposta do servidor os parâmetros de configuração da rede (endereço concedido, máscara de rede, gateway padrão e servidor(es) de DNS).
- Qual o tempo da concessão (lease time)?

## Verificando o aprendizado

Questão 1

**Na atividade envolvendo o protocolo FTP, foi sugerida a seguinte expressão de filtragem do Wireshark:**

```
tcp.flags.syn==1 && tcp.flags.ack==0
```

**O uso dessa expressão facilitou a obtenção da seguinte informação:**

A

Distinguir os endereços de cliente e servidor FTP.

B

O pacote contendo as credenciais de acesso (login e senha) ao servidor.

C

A quantidade de conexões realizadas entre cliente e servidor durante a sessão FTP.

D

Os pacotes contendo os comandos FTP enviados do cliente ao servidor.

E

O tempo de duração de cada uma das conexões entre cliente e servidor.



A alternativa C está correta.

A expressão faz com o que o Wireshark exiba somente o primeiro pacote de cada conexão. Cabe ressaltar que o FTP inicia uma nova conexão para cada transferência. Só pelo primeiro pacote não é possível saber se a conexão TCP foi realmente estabelecida, porém, seu uso é adequado para os fins desta atividade.

## Questão 2

**Na atividade em que analisamos a captura do protocolo TELNET, foi sugerido o uso do recurso “Follow TCP Stream” no Wireshark. Por meio dele obtemos:**

A

A exibição somente dos pacotes relacionados ao protocolo buscado.

B

Todo o tráfego trocado entre cliente e servidor em uma única tela, separados por cores, o que facilita a análise do conteúdo transferido.

C

Dados importantes da conexão TELNET, como o lease time do TCP.

D

A contagem de pacotes distribuída por faixas de tamanho.

E

Os pacotes que foram enviados em duplicidade, destacados na tela do Wireshark.



A alternativa B está correta.

A opção "Follow TCP Stream", selecionada em determinado pacote, faz com que o Wireshark busque todos os pacotes que pertencem à mesma conexão TCP e extraia deles todo o payload, exibindo-o de forma ordenada em uma janela separada. No caso do TELNET, em que o tráfego não é criptografado, é possível observar toda a conversa entre cliente e servidor: comandos, respostas e credenciais (login e senha).

### Considerações finais

Como vimos, o conhecimento dos protocolos de aplicação é de grande importância para todo profissional da área de Tecnologia da Informação. A capacidade de olhar para uma captura de tráfego e dali, entre tantos dados, extrair informações pertinentes é uma habilidade valorizada no mercado.

A análise dos dados transmitidos entre cliente e servidor nos fornece uma grande quantidade de informações acerca da comunicação e, muitas vezes, poderemos identificar a causa de problemas que se manifestam somente nos programas de usuário. Por isso, é tão importante conhecer os protocolos de aplicação, a que se destinam e como funcionam.

Também é importante ter em mente que o domínio das ferramentas usadas neste conteúdo será importante para inúmeras tarefas de diagnóstico, em diversos cenários e com outros protocolos de aplicação.

### Podcast

#### Podcast

Ouçã o podcast sobre protocolos de aplicação da internet.



Conteúdo interativo

Acesse a versão digital para ouvir o áudio.

### Explore+

Pesquise sobre:

- O recurso de COOKIES, usado no protocolo HTTP e veja como ele permite funcionalidades importantes, mas também traz riscos de segurança.
- O protocolo SSH e entenda por que seus recursos de segurança são tão necessários atualmente. Faça um contraste com o protocolo TELNET.
- As bases de gerenciamento (MIB) usadas com o protocolo SNMP e veja como ela trata a questão de gerenciamento de forma abrangente e completa.

### Referências

COMER, D. E. **Redes de computadores e internet**. 6. ed. Porto Alegre: Bookman, 2016.

FOROUZAN, B. A. **Comunicação de dados e redes de computadores**. 4. ed. Porto Alegre: AMGH, 2010.



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

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