

Tecnologia

REDE DE COMPUTADORES

Yanko Costa

# REDE DE COMPUTADORES

Yanko Costa





# **Rede de Computadores**

Yanko Costa



---

Ficha Catalográfica elaborada pela Fael. Bibliotecária – Cassiana Souza CRB9/1501

---

C837r Costa, Yanko

Rede de computadores / Yanko Costa. – Curitiba: Fael, 2017.

296 p.; il.

ISBN 978-85-60531-81-3

1. Redes de computação I. Título

CDD 004.6

---

Direitos desta edição reservados à Fael.

É proibida a reprodução total ou parcial desta obra sem autorização expressa da Fael.

**FAEL**

<b>Direção Acadêmica</b>	Francisco Carlos Sardo
<b>Coordenação Editorial</b>	Raquel Andrade Lorenz
<b>Revisão</b>	Editora Coletânea
<b>Projeto Gráfico</b>	Sandro Niemicz
<b>Capa</b>	Vitor Bernardo Backes Lopes
<b>Imagen da Capa</b>	Shutterstock.com/pixelparticle
<b>Arte-Final</b>	Evelyn Caroline dos Santos Betim

# Sumário

CARTA AO Aluno | 5

1. CENÁRIO DE Redes | 7
2. SIMULAÇÃO DE Redes | 35
3. MODELO OSI e Arquitetura TCP/IP | 65
4. REDE | 91
5. CAMADA DE transporte | 121
6. CAMADA DE Aplicação | 145
7. SERVIDOR HTTP e o Desenvolvimento WEB | 171
8. TRANSFERÊNCIA e Compartilhamento de Arquivos | 199
9. A EMPRESA e Seus Desafios | 225
10. SEGURANÇA EM Rede | 249

CONCLUSÃO | 275

GABARITO | 277

REFERÊNCIAS | 285



# Carta ao Aluno

PREZADO(A) ALUNO(A),

As REDES DE computadores parecem intimidadoras num primeiro olhar. Vários equipamentos, sistemas, cabos, conectores trabalhando em conjunto numa velocidade, muitas vezes, difícil de acompanhar. Mas a medida que são entendidos os conceitos, as regras e métodos utilizados, fica mais tranquilo entender como, apesar do aparente caos, a informação atravessa o planeta em segundos.

Esse livro vem para auxiliar nessa jornada, proporcionando um olhar para a camada abaixo da confusão aparente, mostrando como está estruturada a comunicação entre os dispositivos.

Aos poucos, conjuntos de siglas utilizadas para identificar as tecnologias usadas nas redes são apresentados, permitindo o entendimento e a evolução do conhecimento do leitor nesta importante área da computação.

São apresentados utilitários que permitem observar na prática, como se comportam os sistemas e os protocolos envolvidos na comunicação. Além de solidificar o conhecimento em redes simuladas ou cenários de testes, estas práticas podem ser utilizadas em redes reais para o entendimento, teste e até, em alguns casos, auxiliar no diagnóstico de incidentes que acontecem na rotina de suporte das redes corporativas.

Novas tecnologias e protocolos são constantemente criados, mas precisam coexistir com as grandes redes e sistemas já implantados. O entendimento de conceitos apresentados nesse livro, permitem, facilitar a absorção destas novas tecnologias.

Esse livro não tem a pretensão de esgotar o estudo de redes de computadores. Mas, espero que use este livro como uma ferramenta. Que ele possa contribuir para sua evolução profissional, revendo conceitos, adquirindo novos conhecimentos e testando situações que podem ser implantadas em cenários reais.

Yanko.

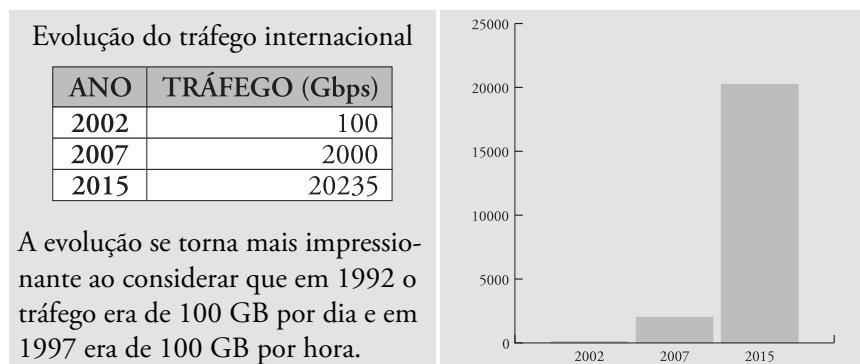
# 1

## Cenário de Redes

NESTE CAPÍTULO, ABORDAREMOS as principais informações sobre como uma rede funciona e como está diretamente ligada a nosso dia a dia dentro e fora da empresa. Apresentaremos os principais componentes e meios de transmissão usados, assim como iniciaremos a estruturação dos conceitos de comunicação de dados.

É IMPORTANTE DESTACAR que o crescimento da quantidade de bytes transmitida pelas redes de computadores tem evoluído de maneira rápida, demonstrando a relevância que esse tópico tem adquirido ao longo dos anos. Podemos observar isso no incremento do tráfego mundial de internet de 2002 a 2015, conforme levantou a Cisco (2016).

Quadro 1.1 – Evolução do tráfego internacional de dados



Fonte: Elaborado pelo autor (CISCO, 2016).

Mas qual é a finalidade das redes? O que tem feito as pessoas insistirem no uso das redes a ponto de incrementar o tráfego dos bytes com essa intensidade?

## 1.1 Uso das redes

O uso das redes de computadores está tão disseminado no dia a dia das pessoas que não é possível executar diversas atividades sem que um dispositivo (computador, smartphone) esteja conectado em rede com outros equipamentos. Temos opções de entretenimento e lazer (jogos, ingressos, pedidos de alimentação), opções de serviços financeiros (transferências, investimentos, pagamentos), serviços relacionados a músicas e filmes online. Em nossa residência, diversos equipamentos se comunicam para compartilhar informações ou permitir o acesso externo a outros serviços. Muitas famílias já possuem acesso banda larga à internet e usam a rede como principal canal de entretenimento, mais do que a própria televisão, se considerar a faixa etária mais jovem.



### Saiba mais

O futuro da TV aberta e a situação das TV por assinatura têm sido postos em discussão devido ao crescimento do acesso a serviços de vídeos online nos últimos anos. Como uma representante de destaque nessa área, segundo informa a UOL, a Netflix no Brasil já

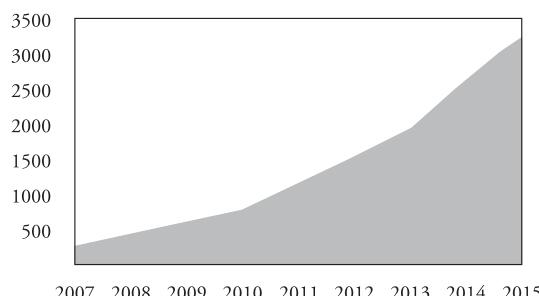
apresenta um faturamento maior que emissoras de TV aberta como Band e Record e, segundo publicações de estatísticas de acesso pelo *Washington Post*, em 2015 o site já usava mais de um terço do tráfego de Internet nos Estados Unidos.

Nas empresas, a dependência das redes é cada vez mais forte. Temos os documentos dos setores centralizados em equipamentos, o acesso a sistemas administrativos espalhados pelos departamentos, as bases de informações financeiras, contábeis, de produção e faturamento integradas por meio da rede interna de computadores.

Tem aumentado o uso de ferramentas de edição de texto compartilhado, nas quais uma equipe pode editar um documento conjuntamente. Esse tipo de estratégia, que utiliza o navegador de sites como base para a interface com o usuário, tem trazido muitos sistemas que antes eram instalados localmente no equipamento do funcionário a serem executados remotamente, até mesmo em servidores fora da empresa. Esse panorama está distribuído em vários tipos de tarefas: sistemas administrativos, sistemas de banco, edição de textos, monitoramento de central telefônica, monitoramento de câmeras de vigilância e até configuração de equipamentos de rede.

O crescimento de assinaturas de conexões fixas de banda larga no mundo inteiro tem sido em média de 60 milhões por ano desde 2005 (ITU, 2016). E o crescimento de assinatura de banda larga em dispositivos móveis tem apresentado um crescimento acentuado desde 2007 (figura 1.1).

Figura 1.1 – Assinatura de serviço de banda larga em dispositivos móveis (em milhões)



Fonte: Elaborada pelo autor (ITU, 2016).

O aceleramento do uso de rede tem implicações na infraestrutura oferecida e no suporte necessário para manter as operações em funcionamento, necessitando de cada vez mais pessoas com conhecimentos das redes e de seus componentes.

Perceberemos, nas próximas leituras, que mesmo o acesso de um smartphone por meio de uma conexão sem fio para o uso da internet, na verdade, refere-se à participação de um equipamento em uma rede.

Na empresa, temos a necessidade de garantir o máximo de disponibilidade da rede, mas também de impedir que pessoas não autorizadas tenham acesso a informações privadas. Essas questões de segurança serão abordadas em mais detalhes em outros capítulos, mas é importante adiantar que, apesar do objetivo de compartilhar informações, em muitos momentos teremos arquivos, imagens e vídeos confidenciais e que requerem privacidade.

Mas o que exatamente é uma rede?

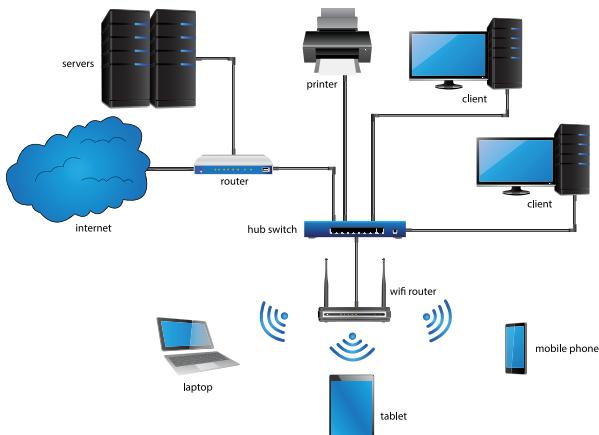
Segundo Tanenbaum e Wetherall (2011), uma rede é “uma coleção de computadores autônomos interconectados por uma única tecnologia”. Mas, a medida que processadores são embutidos em diversos aparelhos (televisão, robôs, câmeras, pontos eletrônicos, consoles de jogos) e estes estão adaptados para comunicação com outros computadores, a rede se torna mais diversificada do que apenas notebooks e desktops. O conceito de IoT – Internet of Things (Internet das Coisas) –, é um exemplo dos itens (sensores, relógios e diversos tipos de aparelhos) que estão sendo desenvolvidos com a característica de conexão em rede e acesso remoto das informações.

Na figura 1.2 verifica-se um exemplo de equipamentos diversos ligados em rede, mas temos em empresas alguns dispositivos que também estão sendo trazidos para as redes de computadores e que antes tinham a própria estrutura de conexão. Um exemplo são as câmeras de vigilância, que tinham um cabeamento específico (análogo) e, com a instalação de câmeras IP (com e sem fio), atualmente compartilham a rede de computadores, até mesmo utilizando uma parte considerável da capacidade da rede pelo fato de transmitir imagens frequentemente. Outros dispositivos são os CLP (controladores lógicos programáveis), que são utilizados para controlar equipamentos em automação industrial e residencial. Esses equipamentos também possuíam tecnologia própria (RS485) para a conexão entre os CLPs e

sensores. Atualmente, estão se comunicando usando a rede de computadores e a arquitetura TCP/IP.

Esses dois casos mostram os esforços da convergência das tecnologias para o uso compartilhado da rede de computadores, reduzindo o custo de instalação, pois aproveitam a estrutura já organizada para a conexão dos computadores e economizam e facilitam a manutenção, já que conhecimentos, acessórios, distribuidores e ferramentas necessárias para dar suporte à estrutura são iguais.

Figura 1.2 – Diversidade de dispositivos a serem ligados em rede



Fonte: Shutterstock.com/Ohmega1982.

Podemos ter uma rede local (LAN – *Local Area Network*), na qual os computadores são conectados uns aos outros participando de um mesmo ambiente próximo. Pode haver interligação entre prédios próximos, andares ou salas comerciais, mas em uma distância relativamente pequena.

Redes com computadores distribuídos em distâncias abrangendo uma cidade são conhecidas como MAN (Metropolitan *Area Network*). A RNP (Rede Nacional de Pesquisa) foi a precursora da internet no Brasil e é responsável pelo projeto Redecomep (Redes Comunitárias de Educação e Pesquisa), que permite a conexão em alta velocidade em regiões metropolitanas de algumas cidades (RNP, 2016).

Nas redes de longo alcance (WAN – *Wide Area Network*), as interconexões podem atravessar oceanos ou grandes territórios de outros países. Um exemplo é a própria internet. Essa comunicação pode se utilizar ainda de cabos submarinos ou satélites para atingir todos os elementos que fazem parte da rede.

As redes também podem se comunicar umas com as outras, grandes ou pequenas, e com diferentes protocolos ou tecnologias. Como resultado, um equipamento dentro de uma rede específica pode se comunicar com outro equipamento em outra rede a vários quilômetros de distância. Esse é um tipo de ligação entre-redes (em inglês, *inter* indica entre e *net*, rede). A internet é um exemplo de conexão entre-redes que tem alcance mundial e um conjunto de protocolos que focaremos e detalharemos nos próximos capítulos, que influenciaram as redes locais. Hoje em dia, esses protocolos são parte das redes locais de empresas e residências.

Para podermos aprofundar nosso conhecimento de como a rede de computadores funciona, vamos identificar alguns de seus componentes. À medida que vamos entendendo uma parte do funcionamento da rede, passamos para um maior detalhamento. Como um microscópio que amplifica e mostra os detalhes dos objetos a cada ajuste das lentes, estaremos aprofundando alguns conceitos das redes conforme a necessidade.

## 1.2 Componentes das redes

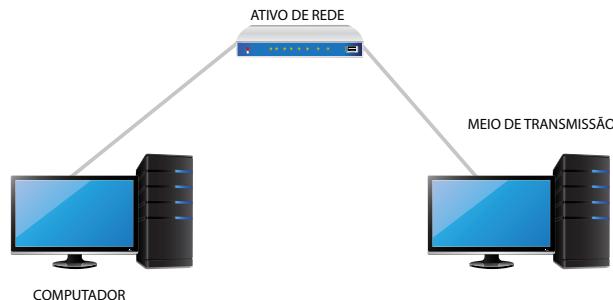
Como componentes da rede, além do computador, temos os dispositivos intermediários, os meios físicos de transmissão e os protocolos. O processo de transmissão de uma informação parte da transformação do bit a ser transmitido em pulso de energia, que usa um meio de transmissão para percorrer o caminho até outro computador.

Como toda a comunicação, há um emissor e um receptor, que podem inverter a comunicação entre eles. Os dois interlocutores seriam os dispositivos finais da rede.

Mas, apesar de poder fazer a comunicação direta entre um computador e outro, normalmente é utilizado um equipamento específico para a comunicação de redes, chamado de ativo de rede, que interligará dois ou

mais computadores entre si. Nesse caso, ele seria um dispositivo intermediário na comunicação.

Figura 1.3 – Componentes básicos



Fonte: Elaborada pelo autor.

Para que a informação seja transferida entre os computadores e os ativos de rede, é necessário criar um caminho entre eles. Esse caminho é denominado meio de transmissão.

## 1.3 Meios de transmissão

Os meios de transmissão representam um caminho que o pulso de informação utilizará para conseguir chegar até outro equipamento. Esse caminho pode ser guiado ou não.

Nos meios guiados, ou seja, com um meio sólido para a passagem da informação, temos o par trançado de cobre, o coaxial e a fibra óptica. Atualmente, nas redes locais, o meio mais utilizado é o par trançado.

### 1.3.1 Par trançado

O par de fio de cobre foi utilizado na rede de telefonia durante décadas e, por esse motivo, foi aproveitado como forma de interligação entre computadores. Dessa forma, estruturas já desenvolvidas tanto para a fabricação quanto para a distribuição desse tipo de cabo puderam ser reaproveitadas para as redes de computadores, diminuindo o custo da instalação e facilitando a expansão das redes.

Os cabos de fio de cobre obedecem à limitação de 100 metros por trecho e as instalações devem ter distância máxima na metragem das instalações. Pelo fio de cobre é enviado um pulso elétrico que atravessa a extensão do cabo até o equipamento receptor, mas um pulso elétrico sempre emite ao redor do cabo uma onda eletromagnética que pode influenciar outro cabo que esteja perto, ainda mais se estiver em paralelo por uma longa distância. Essa influência gerada no outro cabo, no caso da transmissão de dados, é considerada ruído ou interferência, que pode trocar um bit de zero para um e mudar a informação transmitida. Por exemplo: um caractere A na tabela ASCII tem a sequência binária 01000001 representando o número 65. Caso um ruído altere um pulso elétrico e este seja interpretado como um bit 1, dependendo da ordem em que o pulso esteja sendo transmitido na sequência do número, ele pode trocar os bits para 11000001, o que representa o número 193.

Durante as pesquisas com os cabos de cobre, foi descoberto que, ao torcer o fio, a interferência é diminuída. Assim, o termo *trançado* vem da torção feita no par para diminuir a interferência entre os cabos.

Conforme a capacidade máxima da tecnologia utilizada na fabricação dos cabos evoluiu, foram criadas categorias adicionais de cabos, que podiam aumentar a capacidade de transmissão ou melhorar a resistência a ruídos. As categorias mais utilizadas nas empresas atualmente e suas respectivas capacidades estão listadas na tabela 1.1.

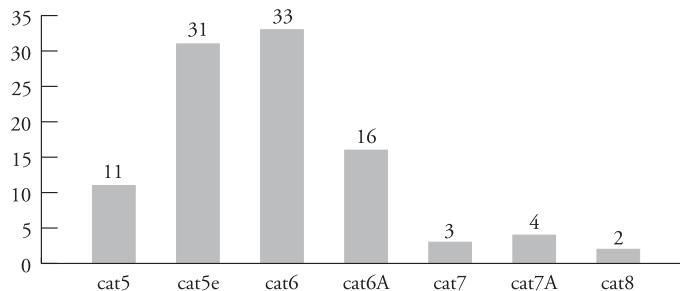
Tabela 1.1 – Principais categorias de cabos em uso

Categorias	Capacidade
5e	1 Gbps
6	1 Gbps
6 <sup>a</sup>	10 Gbps

Fonte: Elaborada pelo autor (COPPER, 2016).

Nos grandes datacenters (locais onde ficam instaladas grandes quantidades de servidores), já é possível observar o uso de categorias mais recentes que ainda estão sendo desenvolvidas. Nesses casos, ao ter equipamentos preparados e maiores capacidades obtidas, o datacenter precisa apenas trocar os equipamentos nas pontas e usar o cabo com a categoria correspondente.

**Figura 1.4 – Categoría de cabos em uso nos datacenters**



Fonte: Adaptada de IEEE (2016).

### 1.3.2 Coaxial

O cabo coaxial permite a transferência de informações a uma alta taxa de velocidade devido à proteção adicional que possui contra interferências. A malha metálica que reveste o fio condutor central diminui a interferência elétrica do caminho. Esse tipo de cabo é muito utilizado para a transmissão de TV a cabo, mas, como as operadoras também aproveitaram a distribuição desse serviço para fornecer conexão à internet, esse tipo de meio voltou a ser usado em conexão com computadores.

### 1.3.3 Fibra óptica

Quando é preciso realizar uma conexão entre equipamentos e existe muita interferência elétrica no caminho, pode ser utilizado um tipo de cabeamento imune a esse tipo de problema, pois utiliza pulsos de luz (laser ou infravermelho) para transmitir os bits. Assim, uma interferência elétrica não surte efeito.

A fibra óptica também permite altas taxas de transmissão e longas distâncias (cerca de 50 quilômetros) sem repetição. Por esse motivo, é utilizada para conectar países e continentes.

As fibras normalmente são utilizadas em pares pois a comunicação mais utilizada é a unidirecional. Dessa forma, para estabelecer uma comunicação entre dois pontos, uma fibra envia e outra recebe o sinal. Como a fibra é

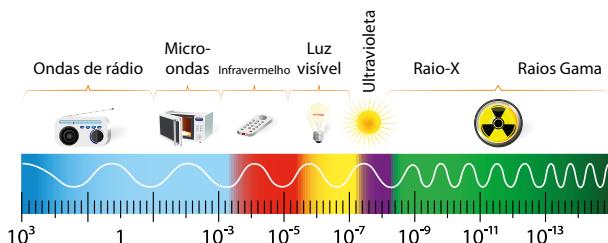
extremamente fina (menor que um fio de cabelo), normalmente os cabos agrupam vários pares de fibras adicionais caso haja ruptura em alguma fibra. Além disso, o custo do processo de instalação é menor. Assim, vale a pena deixar algumas fibras de reserva.

Em alguns casos, onde existe uma dificuldade em passar fibras adicionais, é possível a utilização de diferentes frequências de luz na mesma fibra, tornando a comunicação bidirecional: uma frequência usada para o envio e outra para resposta.

Nos meios não guiados, ou seja, transmitidos pelo ar ou vácuo, há rádio, micro-ondas, infravermelho e laser. A diferença básica entre esses meios é a faixa de frequência no espectro das ondas eletromagnéticas (figura 1.5).

Figura 1.5 – Faixas de ondas eletromagnéticas

## O ESPECTRO ELETROMAGNÉTICO



Fonte: Adaptado de [shutterstock.com/Designua](http://shutterstock.com/Designua).

Nas redes locais, o rádio é o mais presente.

### 1.3.4 Rádio

Quando temos uma área de difícil acesso ou com muitos obstáculos, podemos enviar as informações pelo ar usando as ondas de frequência de rádio. Estações de rádios e televisão já enviam sons e imagens pelo ar, utilizando os mesmos conceitos para enviar bits entre um equipamento e outro. Uma característica importante da transmissão de dados via rádio é que é possível atravessar obstáculos como paredes e móveis. Dessa forma, uma empresa pode implementar a conexão à rede local via rádio entre divisórias, paredes, pisos e outros materiais que compõem a estrutura física da empresa sem que sejam necessárias alterações nas estruturas.

Como veremos mais adiante, os equipamentos utilizados para a transmissão via rádio têm limitações, e mesmo as frequências utilizadas podem ter maior ou menor grau de transposição dos obstáculos sólidos.

Há transmissão de rádio em curta distância e longa distância. Na curta distância, o alcance abrange o domínio de uma rede local (entre uma dezena e algumas centenas de metros). Na longa distância, os equipamentos transmitem dados a dezenas de quilômetros.

### Saiba mais

Padre Landel de Moura foi um personagem brasileiro expoente na comunicação sem fio no Brasil e no mundo, ainda no ano 1892.

Apesar de revolucionário para a época e mesmo com patentes nos Estados Unidos, ficou sem apoio no Brasil e praticamente não é conhecido. Foi um dos primeiros a construir equipamentos para comunicação (de voz e mensagens) sem fio (RODRIGUES, 2016).

#### 1.3.5 Micro-ondas

As ondas eletromagnéticas da mesma faixa dos fornos de micro-ondas podem ser usadas como meio de transmissão para grandes distâncias, porém temos de nos preocupar com a visada, ou seja, é preciso garantir que a onda seja apontada diretamente para o outro ponto e, caso haja algum obstáculo no meio do caminho, é necessário construir torres que possam ficar acima destes.

Também há uma limitação que envolve as grandes distâncias: a occultação do outro ponto se considerarmos a curvatura da Terra. Nesses casos, uma opção é o rebatimento da micro-onda em um ponto da atmosfera que possibilita ainda maior alcance da trans-

Figura 1.6 – Transmissão de micro-ondas via satélite



Fonte: Shutterstock.com/honglouwawa.

missão. Um satélite artificial, posicionado a vários quilômetros, pode ser usado como ponto de rebatimento entre dois pontos.

Mas, para que fosse possível manter estável a comunicação entre os dois pontos, o satélite teria de ficar sempre fixo na atmosfera. Este é exatamente o padrão para os satélites geoestacionários. Quando posicionado a uma distância de 36 mil km, o satélite acompanha a velocidade de rotação da Terra, fazendo que pareça estar fixo em determinado ponto. A grande distância acaba provocando um atraso entre a subida da informação e a descida no outro ponto em cerca de 250 milissegundos.

### 1.3.6 Infravermelho

Em distâncias pequenas e sem obstáculos entre os pontos, pode-se utilizar a faixa de infravermelho. Muitos diferentes dispositivos usam essa faixa, como controles de TV e controles de garagem, mas é importante salientar que esse tipo de frequência não atravessa paredes ou objetos sólidos, ficando confinado em um ambiente.

### 1.3.7 Laser

O sinal laser, além de ser utilizado em fibras ópticas, pode ser usado para a comunicação direta, pelo ar, entre dois pontos. Posiciona-se o equipamento no topo de um prédio e é necessária uma visada direta para os dois equipamentos estabelecerem a comunicação.

Vimos que as estratégias de comunicação ponto a ponto são importantes, mas, para que a comunicação possa ser estabelecida com vários computadores, um ativo de rede auxilia a distribuição dos pulsos, podendo transmitir sinais aos computadores que pertencem à rede local.

### 1.3.8 Placa de rede

A primeira peça de nossa comunicação é a placa de rede, que transforma a informação (bits) em pulsos elétricos para enviar por meio de um cabo de cobre ou em onda eletromagnética para ser irradiada pelo ar.

A informação a ser enviada ou recebida é separada em pequenos trechos chamados de quadros, podendo identificar erros de transmissão a cada

quadro e compartilhar a conexão entre mais computadores, intercalando os quadros das transmissões (COMER, 1998).

O interessante é que a placa de rede permite que a CPU do computador continue com seu processamento normal, sendo interrompida apenas quando o quadro de informação recebido pela rede esteja íntegro (sem erros) e pronto para ser interpretado (KUROSE; ROSS, 2003, p. 300).

Para que vários computadores possam receber e enviar informações via rede, foram criados dispositivos específicos que permitem interligar dois ou mais computadores. Cada dispositivo possui diferentes propriedades e utilização e é importante para entender o real funcionamento de uma rede, seja residencial, seja corporativa.

Uma rede corporativa pode ter apenas um ativo, um conjunto de ativos ou uma combinação de diferentes ativos ou até mesmo todos os ativos descritos. Essa complexidade, juntamente à utilização de vários protocolos de comunicação, como será visto mais adiante, é que revela a importância do profissional de redes dentro de uma empresa. Normalmente o diagnóstico de um problema de interrupção na rede envolve o conhecimento sobre o funcionamento dos vários equipamentos e a forma de utilização dentro do cenário próprio da empresa.

### 1.3.9 Hub

O dispositivo mais utilizado durante a implantação das primeiras redes foi o hub, que concentra o cabeamento das redes e permite que várias máquinas conversem entre si. Para isso, possui várias entradas para a inserção de plugues de conexão (conectores) chamadas portas. Esses plugues padronizados nas redes de fios de cobre têm um formato retangular com o qual são prensados os fios, chamados de RJ45.

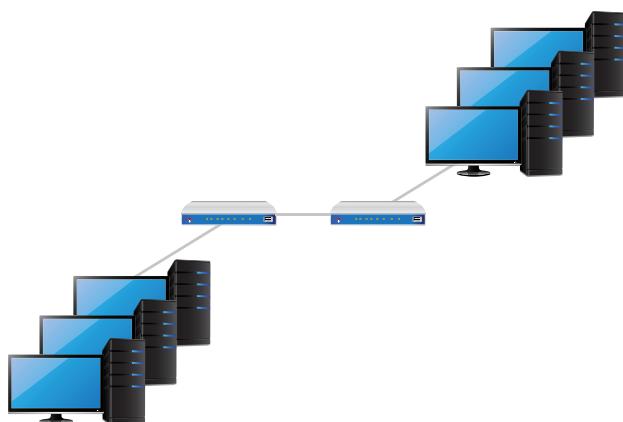
Uma característica dos hubs é a repetição da informação de cada máquina para todas as portas do equipamento, também chamada transmissão por difusão. Dessa forma, todas as máquinas conseguem identificar a informação que está sendo transmitida entre dois equipamentos. Outra característica importante é que o total da capacidade de transmissão, por ser compartilhada, é dividido por todas as máquinas conectadas.

## Rede de Computadores

À medida que as redes crescam, as características do hub limitavam o uso da rede quando a quantidade de equipamentos fosse grande e houvessem mais hubs interligados. Imagine um hub com 20 portas conectado a outro hub também com 20 portas. Se o primeiro estivesse compartilhando a banda de 10 Mbps com 20 equipamentos, um vigésimo da capacidade estaria sendo usado para conectar o segundo hub. Este, por sua vez, estaria compartilhando novamente a conexão com os outros 20 computadores, prejudicando muito a transmissão das informações na rede inteira.

Mesmo quando uma grande rede necessitasse interligar vários hubs, era preciso ter cuidado com o tempo que um quadro era transmitido entre dois dispositivos que estivessem em cada ponta da rede. Devido ao protocolo Ethernet, de interromper a transmissão ao perceber que algum dispositivo já está utilizando o meio físico para que não aconteça uma colisão, era necessário avaliar quantos ativos e trechos de cabo o quadro percorria. Um limite usado como regra de projeto de redes indicava que deveriam ser dispostos até quatro ativos interligados em série.

Figura 1.7 – Interligação entre hubs



Fonte: Elaborada pelo autor.

Para evitar perda de capacidade em uma grande rede, foi desenvolvido um novo tipo de ativo com características bem diferentes do hub, mas com a mesma possibilidade de concentração da comunicação entre as máquinas: o switch.

### 1.3.10 Switch

O switch permitiu o aumento da rede sem que a capacidade de transmissão fosse dividida entre as máquinas. Outra mudança é que a informação entre dois equipamentos não é repetida entre todas as portas do equipamento, o que aumenta a segurança de uma rede, uma vez que a possibilidade de uma máquina interceptar a comunicação de outras duas é mais difícil.

Para conseguir o isolamento dos dois equipamentos dos demais dispositivos da rede, o switch tem uma tabela interna na qual mantém um registro de todas as máquinas que estão conectadas a ele. Quando uma máquina precisa se conectar a outra na mesma rede local, o switch verifica em qual porta esses dispositivos estão conectados e estabelece uma comunicação direta entre as duas portas sem que seja compartilhada internamente a comunicação. Ao interligar as duas portas, o switch realiza a chamada comutação de circuitos das duas portas.

Apesar do aumento de capacidade proporcionado pelo switch, até o momento, estamos focados na rede local, ou seja, os equipamentos fazem parte do mesmo ambiente de transmissão. Mas, com o passar dos anos, surgiu a necessidade de comunicar a rede local com outras redes locais de empresas e organizações diferentes. Nesse caso, saindo da rede local e partindo para a interligação entre redes, há a necessidade de encaminhar a informação para outro equipamento que não está conectado na mesma configuração lógica de rede. O processo de encontrar o caminho da outra rede com a qual desejamos nos comunicar é chamado de descobrimento de rota ou roteamento, e o equipamento responsável por esse algoritmo é o roteador.

### 1.3.11 Roteador

O roteador tem esta característica: inspeciona o pacote de informação e, por meio de tabelas e algoritmos de roteamento, consegue encaminhar a informação para outra rede. Ele interliga uma ou mais redes e repassa a informação diretamente para a rede de destino ou para outro roteador que tentará identificar qual será o caminho necessário para chegar até a rede de destino final. No início dessa tecnologia de interligação de redes, apenas o roteador executava a função de roteamento. Mas os switches também começaram a incluir a possibilidade de inspeção dos pacotes e avaliar se pertenciam à mesma rede ou deveriam ser encaminhados para outra rede.

A interligação com outras redes pode ser conseguida também com um computador e duas ou mais placas de redes. Com o software apropriado, o computador pode calcular para qual interface de rede ele deve encaminhar o pacote, da mesma maneira que o roteador. Essa, inclusive, é a base de funcionamento dos firewalls: antes de encaminhar o pacote para a rede solicitada, verificam se existe alguma condição que indicará se filtrará e bloqueará a comunicação.

Até agora, nos cenários de hubs, switches e roteadores, estávamos relacionados a uma rede em que o meio de transmissão envolvia algum tipo de fiação. Mas, como vimos nos tópicos dos meios de transmissão, atualmente nas empresas e nas residências temos também a possibilidade de enviar a informação pelo ar. Muitas operadoras de telecomunicações já instalam um equipamento híbrido que pode compartilhar a rede via rádio, além de estabelecer a comunicação com a rede principal. Esse dispositivo é o Access Point e pode ser combinado com switches e modems ao ser instalado em uma residência.

### 1.3.12 Access Point

Este dispositivo recebe a informação via fiação e transfere o pulso elétrico por meio de uma antena que irradia a informação via onda eletromagnética na faixa da frequência de rádio. Muitos equipamentos desse tipo acabam sendo agrupados a outras funcionalidades por questões de custo e praticidade. É comum vermos no mercado um equipamento que reúne as características de roteador, switch e Access Point. Como vêm pré-configurados de fábrica, o próprio usuário pode conectar o aparelho seguindo as instruções básicas e usar para ampliar a abrangência da rede local e permitir maior mobilidade no acesso.

Com relação às limitações da tecnologia, o Access Point tem as mesmas características do hub: compartilha a faixa da banda entre os equipamentos e repete a informação para todos os dispositivos conectados. Também está limitado à distância em que o sinal ainda tem potência suficiente para ser acessado. A quantidade e os tipos de obstáculos sólidos no caminho influenciam nessa distância. Um campo aberto, barracão ou área externa permite que o sinal ultrapasse centenas de metros. O uso de antenas mais potentes também amplia a distância que esses aparelhos podem atingir.

Atualmente, com a disseminação dos aparelhos sem fio nas residências, vem sendo utilizada uma extensão do Access Point chamada repetidor. Esse aparelho vem pré-configurado para acessar o Access Point principal e amplificar o sinal para uma parte da residência em que o sinal não chega muito forte, aumentando a área de cobertura. Esse mesmo tipo de funcionalidade pode ser feito com mais de um Access Point instalado em um local onde há bom sinal do aparelho principal.

Softwares abertos, como o “dd-wrt”, podem ser instalados em aparelhos Access Point para que tenham mais funcionalidades ou atualização de tecnologias que o fabricante original do aparelho não disponibiliza.

---

## 1.4 Servidores

Ainda é preciso destacar qual é o propósito de todas essas tecnologias. Afinal, qual é o objetivo da evolução das conexões entre os equipamentos? A verdade é que temos a necessidade de compartilhar recursos e informações; temos a necessidade de prestar serviços a distância para clientes; e desejamos nos comunicar com familiares e amigos quando estivermos em movimento. Para todas essas opções, precisamos de equipamentos que permitam que vários usuários possam acessar a rede ao mesmo tempo e processar a informação solicitada, criando condições para que várias máquinas transfiram arquivos com filmes, dados, imagens, voz, planilhas ou apenas pequenos textos.

Esses equipamentos são chamados de servidores. Normalmente o grande foco da conexão dos computadores pessoais ou das empresas é, apesar de qualquer máquina poder compartilhar recursos e informações com o software adequado, quando se avalia o volume estimado de acesso, verifica-se a necessidade de equipamentos específicos para essa situação. As características dos servidores demandam também um tipo de hardware que é

fabricado considerando a possibilidade de que vários usuários accessem as informações simultaneamente.

Softwares preparados para receber um comando via rede, preparar a informação (independente do formato) e transmitir de volta como resposta são desenvolvidos para realizar essas atividades com o máximo de desempenho possível. Por exemplo: placas de rede com capacidade de processamento prévio de pacotes de rede, discos de alta capacidade para o armazenamento do resultado de várias conexões, placas-mãe com vários processadores na mesma máquina podendo processar em paralelo as demandas.

O sistema operacional dos servidores também precisa estar preparado para a função, e dois sistemas se destacam nessa categoria: os sistemas GNU/Linux e os sistemas Windows Server.

Os sistemas GNU/Linux são muito utilizados como sistema base para os dispositivos de redes (switches, roteadores e Access Points) e para os servidores encarregados de grandes volumes de acesso (Facebook, Google, Twitter, Amazon). Esses sistemas são baseados em padrões do Unix, que, por sua vez, vem sendo utilizado em equipamentos de alto desempenho nos últimos 40 anos. Sua arquitetura foi desenvolvida para ser multitarefa e multiusuário e foi nesse sistema que tivemos as primeiras implementações do protocolo TCP/IP, detalhado nos próximos capítulos.

O Windows Server é uma presença constante em empresas em que há grande volume de equipamentos desktop e também nas pequenas empresas. Tem uma arquitetura preparada para ser multitarefa e também capacidade de resposta para várias conexões simultâneas, o que o habilita para a tarefa de compartilhamento de arquivos de informações.

Os dois sistemas possuem interface gráfica e permitem a execução de comandos por meio de janelas de comando. Mas o Windows tem um foco voltado ao uso de janelas gráficas em sua arquitetura, e o Linux acaba usando os sistemas gráficos como um complemento de seu sistema. Por esse motivo, o Linux tem grande quantidade de utilitários preparados para serem utilizados em linha de comando. Sem falar que esses utilitários podem ser combinados uns com os outros e usados em conjunto com uma linguagem de scripting (shell), permitindo a automação de tarefas repetitivas.

Será utilizada para os exemplos neste e nos demais capítulos a distribuição GNU/Linux Mint 17 Qiana. Em alguns trechos, alguns comandos podem ser comparados com o Windows 10 para exemplificar as semelhanças.

### 1.4.1 Utilitários e linha de comando

Considerando que um servidor estará executando ações por meio da rede e que uma interface gráfica (GUI – Graphic User Interface) ocupa um espaço considerável do tempo do processador e da memória do servidor, o uso de interfaces de linha de comando (CLI – Command Line) permite economizar diversos ciclos de processamento no equipamento. Nesse caso, um sistema com menos recursos, mas sem interface gráfica, pode ter um desempenho melhor que um sistema com mais recursos, porém com o uso constante da interface gráfica.

O Linux possui diversas ferramentas utilitárias para testes e diagnóstico de redes e por isso será utilizado como base em vários exemplos. Esse sistema é livre e pode ser utilizado sem interface gráfica, o que facilita o uso em equipamentos com pouca capacidade ou virtualizados.

---

---

Para fins de estudo, é possível criar virtualmente uma pequena rede e testar os utilitários que serão apresentados nos diversos capítulos. Softwares de virtualização permitem executar um sistema operacional em uma janela do sistema operacional normal. Ou seja, se estiver usando Windows, pode executar um sistema Linux ou Mac OS dentro de uma janela e usá-lo normalmente. Juntamente com a execução do sistema operacional, o virtualizador permite criar conexões entre sistemas como se estivessem em uma rede local. Dessa forma, podem ser feitos testes vendo a resposta de um sistema real. Um dos softwares é o VirtualBox, que pode ser instalado tanto em Linux quanto em Windows. Ele simula um equipamento (hardware) no qual podemos instalar um novo sistema e

permite configurar placas de redes adicionais para testes.

Instale dois Windows, dois Linux ou então um Windows e um Linux e teste a comunicação entre ambos usando os utilitários que mostraremos nos próximos capítulos.

Existem softwares virtualizadores de servidores também, que abordaremos nos últimos capítulos, mas têm características diferentes e abordagem mais profissional.

---

A maioria das ferramentas é utilizada no ambiente de linha de comando por serem mais práticas e fáceis de reproduzir (a partir do Windows 10, também está sendo adotado o mesmo ambiente de linha de comando padrão do Linux, apesar de nem todos os utilitários estarem disponíveis). Mesmo que seja usada a linha de comando padrão do Windows, alguns utilitários são semelhantes, pois foram incorporados ao DOS (sistema ancestral do Windows), vindos do Unix (sistema ancestral do Linux).

Uma primeira atividade em um computador é reconhecer as interfaces de redes e suas configurações. Estando na janela da linha de comando execute *ipconfig* no Windows ou *ifconfig* no Linux e terá um resultado semelhante ao mostrado na figura 1.8.

## Saiba mais

Pesquise como abrir a área da linha de comando no sistema que estiver usando. No Windows, o arquivo executável é *cmd.exe*, e em muitas versões existe um campo para executar esse programa no menu Iniciar. No Linux, diversas distribuições têm um executável chamado *terminal*, que abre uma janela com a linha de comando.

Ao executar o comando, não se preocupe em ter um resultado um pouco diferente da Figura 1.8. Esses comandos mostram a configuração de rede em tempo real de cada máquina e assim podem mostrar quantidades diferentes de interfaces. Por exemplo: a listagem do *ifconfig* do Linux mostra quatro

interfaces de redes: uma interface de rede com fio (eth0), uma interface de teste (lo), uma interface virtual (virbr0) e uma interface sem fio (wlan0).

Em cada capítulo, conhceremos mais sobre essas informações de configuração, como um computador encontra o outro em qualquer parte do mundo e como levantar dados para identificar problemas quando não é possível estabelecer a conexão.

Figura 1.8 – Linha de comando Windows e Linux com a lista das interfaces de rede



Fonte: Elaborada pelo autor.

Também com esses utilitários temos a possibilidade de analisar estatísticas de uso da rede. Na janela do Linux, é possível verificar informações de pacotes transmitidos (TX) e recebidos (RX) pela interface no próprio resultado do comando *ifconfig*. Na linha final de cada interface temos o resumo em bytes de quanto essa interface transmitiu e recebeu desde que o computador foi iniciado. Por exemplo: (wlan0) RX bytes:11216337519 (11.2 GB) TX bytes:247534913 (247.5 MB).

Para ter as mesmas informações estatísticas de quanto a interface transmitiu e recebeu no Windows, temos de usar o comando *netstat* com a opção *-e*.

Figura 1.9 – Estatística de interface sem fio no Windows 10

Estatísticas de interface		
	Recebido	Enviado
Bytes	1753486406	157226976
Pacotes unicast	4744212	1783602
Pacotes não unicast	82080	19794
Descartados	0	0
Erros	0	0
Prot. desconhecidos	0	0

Fonte: Elaborada pelo autor.

O comando *netstat* é também encontrado no Linux e apresenta diversas outras informações do tráfego feito pela placa de rede do equipamento. Essas informações podem ser segmentadas em protocolos, usuários e até no status da comunicação.

A possibilidade de se ter mais de uma placa de rede (como vimos na seção sobre roteadores) pode acontecer com os servidores e faz que atue em mais de uma rede diretamente sem passar por nenhum outro dispositivo para fazer essa conexão. Os sistemas operacionais conseguem fazer a verificação para qual placa de rede a informação deve ser enviada. Assim, uma base de dados ou um serviço de compartilhamento de arquivos pode responder para mais de uma rede, avaliando de qual placa de rede recebeu a solicitação e enviando a resposta pelo mesmo caminho. Essa abordagem, porém, acaba criando mais de um caminho para os pacotes saírem de uma rede, podendo levar a algum engano no diagnóstico de possíveis problemas. No aspecto de segurança, tem-se mais uma porta de entrada de vírus e malwares para a rede interna.

---

Caso não haja um Linux instalado na máquina local, é possível acessar remotamente a linha de comando de outro Linux que esteja na rede ou na internet. Para isso, existem utilitários que acessam o Linux remoto via protocolo SSH e permitem que, usando um usuá-

rio com senha e indicando o endereço da máquina, seja disponibilizada uma janela com a linha de comando. Nesse caso, as informações visualizadas pertencem ao computador remoto e cada comando executado será processado também no computador remoto. Esse acesso pode ser feito de Windows para Linux (pode ser utilizado o Putty como utilitário no Windows) e de Linux para Linux com o comando SSH na linha de comando.

---

Outros utilitários serão mostrados assim que tivermos mais conhecimento dos protocolos nos próximos capítulos.

#### 1.4.2 Pacotes e endereços

As ferramentas vistas permitem ilustrar também um importante conceito na transmissão de informações via rede: o pacote. Para não ocupar o meio de transmissão durante todo o envio de um arquivo, as informações são separadas em pequenos trechos (pacotes) e enviadas separadamente. Mais de um computador pode transmitir uma informação se os pacotes forem intercalados. Para que possamos identificar cada pacote e sua origem, seu destino e sua sequência, temos protocolos que fazem a organização, os quais veremos mais adiante com detalhes.

Neste momento, é importante saber que uma conexão de rede é estabelecida apenas com um endereço numérico, apesar de ser comum lembrar dos endereços de texto que são usados para nomear os sites que possuem conteúdos diversos. Esses endereços de texto são chamados de domínios e precisam ser convertidos para os endereços numéricos, senão a comunicação não é estabelecida.

Também é comum criar nomes para facilitar a identificação dos servidores e dos demais dispositivos, mas, da mesma maneira, é preciso fazer a conversão para um endereço numérico, senão as máquinas não são encontradas. Esse mecanismo de endereçamento será visto com detalhes nos próximos capítulos.

## 1.5 Cabeamento

Como a conexão entre computadores usando fio de cobre é uma das mais utilizadas e o fio de cobre também é usado na comunicação de telefonia (e ramais internos), o compartilhamento dessa faixa permite economizar na infraestrutura utilizada para a passagem dos cabos, além de trazer flexibilidade no uso dos fios. Essa forma de compartilhamento é chamada de cabeamento estruturado.

O compartilhamento, apesar do uso do mesmo tipo de cabo, precisa ser previamente planejado, pois os equipamentos que conectam computadores são diferentes e utilizam uma sinalização elétrica diferente das centrais telefônicas (PABX) que interligam os ramais.

Alguns acessórios podem ser utilizados para organizar o cabeamento e tornam possível a utilização conjunta de parte do cabeamento entre ramais e computadores. Não são ativos, pois não atuam na comunicação, ou seja, não modificam o pacote com a informação, apenas organizam as conexões. No Quadro 1.2, temos a descrição dos principais acessórios.

Quadro 1.2 – Acessórios utilizados para organização de cabos

PATCH PANEL	Quadro no qual os cabos são conectados à parte traseira em um conjunto de conectores fêmeas, ficando a ligação disponível para ser usada para interligar um equipamento, outro patch panel ou voice panel.
VOICE PANEL	Semelhante ao patch panel, mas apropriado à conexão de fios de telefonia, no qual são conectados a um conjunto de conectores fêmeas e a ligação dos cabos fica disponível para ser usada para interligar um equipamento, outro patch panel ou voice panel.
DIO	Uma fibra é aberta e seus pares ficam conectados a um conjunto de conectores ópticos, sendo possível sua ligação posterior a um equipamento.

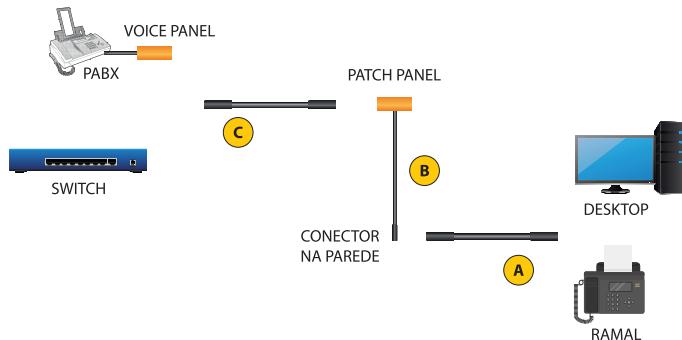
PATCH CORD	Um cabo com dois conectores nas pontas, funciona como um elo de ligação. Pode utilizar fibra óptica ou fio de cobre (UTP). Esse acessório será utilizado para conectar um patch panel ou patch voice com os equipamentos. Também é utilizado para ligar os computadores aos espelhos com conectores fêmeas que ficam instalados nas paredes das salas. Muitos fabricantes de computador já enviam um patch cord ao entregar os equipamentos.
TRANSCEIVER	Dispositivo utilizado para transmitir e receber sinal. Pode ser confeccionando com um complemento a um ativo de rede. Um switch pode receber um transceiver de fibra óptica e assim se comunicar diretamente com a fibra.
CONVERTER	Dispositivo usado para converter um tipo de sinal para outro. Em redes, temos como exemplo a necessidade de converter o sinal de um cabo UTP Ethernet para conectar em um cabo de fibra óptica.

Fonte: Elaborado pelo autor.

Na figura 1.10, podemos visualizar um esquema de interligação no qual uma parte da estrutura (calhas, canaletas, cabos) pode ser utilizada tanto para a conexão de ramais quanto de computadores. No diagrama, vemos o item **A**, que é um cabo UTP com um RJ45 em cada uma das duas pontas (chamado de patch cord). Esse cabo conectará um computador ou um ramal. Na parede do recinto, item **B**, existe um plugue RJ45 fêmea que está conectado diretamente ao painel organizador de cabos (patch panel), que é um grupo de RJ45 fêmeas. Nesse caso, com o item **C** é possível fazer uma conexão usando novamente um patch cord entre um switch e um patch panel. Também é possível estabelecer uma conexão entre o patch panel e um voice panel (semelhante a um patch panel, mas usado para distribuir os fios telefônicos que saem de um PABX e vão para os ramais).

## Rede de Computadores

Figura 1.10 – Esquema de compartilhamento de cabos entre ramais e computadores



Fonte: Elaborada pelo autor.

Logicamente, se um local precisar de um ramal e de um computador conectados em rede, será necessário passar dois cabos entre o patch panel até o local em que estarão os dois aparelhos. Será aproveitada toda a tubulação e a passagem de cabos (item **B**) entre os dois pontos. Um dos patch cords conectará o ramal e o outro conectará o computador. No item **C**, teremos um cabo conectando o PABX e outro, o switch.

Normalmente já se prevê um cabo adicional para cada local, assim pode ser usado para mais um computador ou para mais um ramal.

## Síntese

Vimos neste primeiro capítulo como as redes estão inseridas no ambiente e nas atividades de várias pessoas. Muitas tarefas rotineiras, como acessar a conta bancária, assistir a um filme e agendar uma passagem podem ser feitas a distância. Inclusive o pagamento desses serviços.

Para começar a entender as redes, pudemos identificar equipamentos que facilitam a conexão na rede local (hubs, switches, Access Point) e entre redes (roteador). Nessa comunicação, temos várias formas de transferir os bits de um local para outro: usando eletricidade por meio de fios de cobre, luz via fibra óptica e ondas eletromagnéticas enviadas pelo ar.

Utilitários nos sistemas operacionais podem auxiliar a identificar as interfaces e as estatísticas da comunicação feita pelos computadores.

Para facilitar a administração dos cabos e a infraestrutura necessária para distribuí-los, podemos ter um compartilhamento de cabos com os aparelhos de telefonia existentes nas empresas realizando a convergência das redes no cabeamento estruturado.

## Atividades

1. Há diferentes tipos de redes em uso, conforme a abrangência geográfica. Interligue abaixo alguns exemplos de rede e conexões com suas classificações.
  - (1) MAN (Metropolitan Area Network)
  - (2) LAN (Local Area Network)
  - (3) WAN (Wide Area Network)

( ) Redes que interligam dispositivos usando cabos submarinos e satélites entre dois países.

( ) Conecta equipamentos distribuídos na mesma sala.

( ) Equipamento ligado a um servidor em outro bairro da cidade.
2. Uma empresa possui dez equipamentos ligados em rede por um hub. Se investir na troca do equipamento por um switch, cite e descreva uma vantagem dessa substituição.
3. Uma empresa tem diversos arquivos utilizados pelos funcionários internamente para a execução de tarefas. Os arquivos que são acessados pela rede local da companhia ficam armazenados em qual equipamento? No roteador ou no servidor? Indique em qual e explique a diferença entre os dois.
4. O diretor de uma empresa tem um notebook e participa de várias atividades e reuniões dentro da companhia. Em todos os momentos, precisa avaliar as informações e os dados do sistema administrativo para tomar decisões. Qual equipamento de rede pode ser utilizado para facilitar a mobilidade do diretor?



# 2

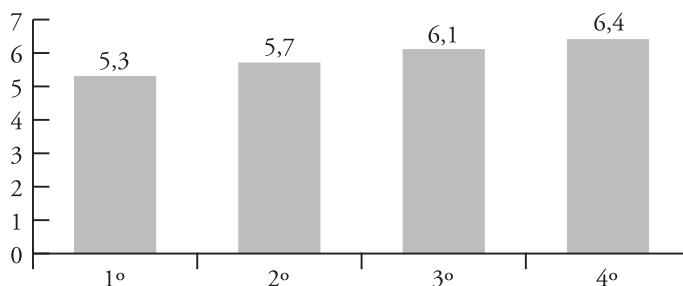
## Simulação de Redes

NESTE CAPÍTULO, ABORDAREMOS o protocolo Ethernet, seu formato e seu funcionamento. Utilizaremos ferramentas de simulação para iniciar o projeto e a prática de alguns cenários de redes e entender seu funcionamento. Apesar de ainda não termos abordado todos os conceitos e protocolos necessários para o completo funcionamento de uma rede, o conhecimento desses simuladores permitirá o desenvolvimento de testes práticos. A partir deste capítulo, iremos utilizar mais funcionalidades desses softwares à medida

que forem detalhadas mais dessas tecnologias. Também veremos mais utilitários de diagnóstico de rede diretamente no sistema operacional, os quais permitirão observar como as redes estão configuradas e auxiliar na resolução de problemas.

O protocolo Ethernet vem sendo utilizado em redes locais desde a década de 1970. Desenvolvido por Bob Metcalf e David Boggs enquanto estavam na Xerox, o protocolo foi padronizado posteriormente pela IEEE. Seu crescimento é constante, como podemos comprovar pelo mercado de vendas de Switch Ethernet em 2015 no gráfico a seguir (figura 2.1).

Figura 2.1 – Crescimento do mercado de *switches*



Fonte: Elaborado pelo autor com base nos dados da IDC (IDC, 2016).

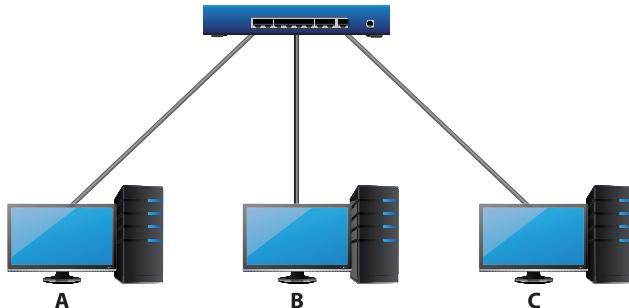
Entre os anos de 2014 e 2015, houve um aumento de 1,9% (IDC, 2016). O uso de Ethernet está disseminado pelas empresas, e as redes locais estão basicamente estruturadas por meio dela. Nos próximos capítulos, veremos também como usar o protocolo Ethernet em conjunto com outros protocolos de rede.

## 2.1 Ethernet

Como já vimos, um computador em uma rede local precisa de uma placa de rede, um meio de transmissão (por exemplo o cabo UTP) e um ativo de rede (um *switch*) para facilitar a troca de informações com mais de um dispositivo. Nesse caso, considerando nossa visão sobre a rede em termos de hardware, estaríamos prontos para transmitir uma informação de

um computador para outro. Contudo, vamos imaginar a situação da rede seguinte (figura 2.2).

Figura 2.2 – *Switch* com três computadores



Fonte: Elaborado pelo autor.

Na figura 2.2, a máquina **A** vai enviar uma informação para a máquina **C**. Como a máquina **C** detecta a informação (pacote) como enviada para ela? Como o switch vai identificar que deve enviar a informação para a porta que a máquina **C** está conectada? Lembre-se de que vimos que o switch (diferentemente do *HUB*) envia o pacote diretamente para a máquina que está conectada em suas portas e não para todas as portas e, para isso, ele comuta (interliga) o circuito das duas portas.

Esse trabalho de estruturar os bits na saída da placa de rede e na chegada, dividir as informações em pequenos trechos, organizar a sequência e estabelecer uma forma de identificar cada máquina são características de um **protocolo**. Ou seja, o protocolo é um conjunto de regras para comunicação: o modo como os itens da rede devem separar, organizar, identificar e enviar os bits de maneira padronizada para que outras máquinas que se organizarem sob o **mesmo procedimento** possam entrar na comunicação e reconhecer a mensagem.

Existem vários protocolos envolvidos na conexão entre um equipamento e outro. Cada protocolo acaba executando uma tarefa específica para que fique mais flexível quando as tecnologias envolvidas são trocadas, alteradas ou evoluídas. Assim, a mudança em um protocolo tem menor impacto nos outros. No caso, vamos verificar um protocolo que é utilizado para controlar a parte física da rede: o protocolo **Ethernet**. Nesse nível de interligação

das placas de redes e cabos, o Ethernet cuida para estabelecer uma forma de identificar cada integrante da rede (*endereço*), uma forma de conferir se os bits enviados estão corretos e uma forma de detecção se o meio físico está pronto para o uso. O padrão está relacionado também com características elétricas dos cabos, distâncias máximas, nível de ruído aceitável e formas de encaixe entre os dispositivos.

## 2.2 IEEE 802

Com seu sucesso, o protocolo Ethernet foi redirecionado para um grupo de padronização internacional (IEEE), que alocou os padrões relacionados ao Ethernet sob o número 802.3. Nesse grupo, 802 está relacionado aos padrões dedicados às redes locais, e cada seção está dedicada a uma característica ou um novo tipo de protocolo. Temos, por exemplo, na seção 802.11 as informações relacionadas às redes sem fio. Alguns padrões estão obsoletos (como o 802.5 – *Token Ring*), outros foram acrescentados ao grupo conforme a tecnologia de redes foi evoluindo.

Na prática, o IEEE seguiu as mesmas considerações do Ethernet desenvolvido na Xerox e posteriormente padronizado em conjunto com a DEC e Intel (chamado também de padrão DIX). Por esse motivo, pode ser chamado tanto de padrão Ethernet ou padrão IEEE 802.3, como referência ao entendimento do protocolo. O IEEE introduziu uma pequena modificação no formato do pacote Ethernet no espaço reservado ao tipo de protocolo para que se pudesse estabelecer o tamanho do pacote. Mas o padrão anterior estava implementado em muitos equipamentos e, para não criar um problema, foi considerado que esse campo poderia ter tanto o tamanho como o tipo de protocolo. Para isso, bastaria comparar o valor utilizado nesse campo com o valor hexadecimal 0x0600. Conforme o valor encontrado, se for acima desse valor, deve-se pular os 8 bytes acrescentados pelo 802.3 na sequência do pacote. Dessa forma, o padrão original Ethernet continuou sendo utilizado nos equipamentos de rede.

Usaremos o termo Ethernet durante todos os capítulos, ele se refere tanto ao padrão Ethernet II (DIX) quanto ao 802.3 da IEEE.

## 2.3 Pacote Ethernet

Para iniciar, vamos precisar de um endereço, uma forma da máquina interpretar o pacote como sendo para ela e para saber qual será o destino da informação que irá enviar.

O protocolo Ethernet, de modo geral, identifica a origem e o destino do pacote, verifica se houve alguma alteração durante a transmissão da informação e insere a informação em seu quadro.

A seguir, temos a estrutura básica do Ethernet.

Quadro 2.1 – Estrutura do pacote Ethernet

Preâmbulo	Endereço de destino	Endereço de origem	Tipo / tamanho	Dados	CRC
8 bytes (64 bits)	6 bytes (48 bits)	6 bytes (48 bits)	2 bytes	46 – 1500 bytes	4 bytes

Fonte: Elaborado pelo autor com base em Stevens (1992).

Conforme aponta Comer (1998, p. 33) o campo *preâmbulo* inicial apenas formata uma série de bits intercalados para sincronizar as placas de rede. Os demais campos estabelecem a estrutura do quadro que identificam como a informação deve ser recuperada.

### 2.3.1 Endereço Ethernet

O Ethernet usa dois campos de 6 bytes (48 bits) da sua estrutura para armazenar o endereço de origem e destino do pacote. Esse endereço, comumente chamado de *MAC address*, utiliza a seguinte notação: 6 números hexadecimais separados por “:”. E os três primeiros bytes indicam o fabricante da placa de rede. Esse identificador é conhecido como OUI (*Organizationally Unique Identifier* – ou, numa tradução livre, identificador único organizacional). Os outros três números hexadecimais indicam um número sequencial único que o fabricante irá utilizar em cada dispositivo que distribuir.

Por exemplo, numa placa de rede que tenha o endereço Ethernet 80:29:94:E1:EE:0F, os três primeiros bytes (80:29:94) podem ser pesquisados na base de dados pública do IEEE para identificar o fabricante, como na figura 2.3 seguinte.

Figura 2.3 – Pesquisa de fabricante na base de fabricantes

The screenshot shows a search interface for IEEE MAC addresses. The search bar contains the value '802994'. Below the search bar are buttons for 'Filter' and 'Reset'. A dropdown menu says 'View 10 rows'. To the right, it shows 'Showing 1 - 1 of 1'. The main table has a header 'MAC ADDRESS BLOCK LARGE (MA-L) SEARCH RESULTS' with columns: Assignment, Type, Company Name, and Company Address. One row is displayed: Assignment is '80-29-94 (hex)', Type is 'MA-L', Company Name is 'Technicolor CH USA Inc.', and Company Address is '101 West 103rd St, Indianapolis IN 46290 US'.

Assignment	Type	Company Name	Company Address
80-29-94 (hex)	MA-L	Technicolor CH USA Inc.	101 West 103rd St, Indianapolis IN 46290 US

Fonte: IEEE (2016).

Com esse esquema, é possível ter apenas um único endereço *MAC address* ativo por dispositivo numa rede local, e assim conseguimos identificar cada dispositivo de rede unicamente. Quando um computador envia uma informação, insere no quadro Ethernet o endereço de destino (*MAC address* da máquina que vai receber o pacote) e o endereço de origem (*MAC address* da própria placa de rede).

Se o ativo de rede for um *HUB*, todas as portas vão receber uma cópia do quadro, mas apenas a máquina que possui o *MAC address* informado no campo de endereço de destino irá aceitar o pacote e utilizar seu conteúdo de dados.

Num *switch*, o pacote será verificado, e o equipamento irá identificar qual máquina está ligada em alguma de suas portas e tem o *MAC address* de destino e encaminhar apenas para essa porta todos os pacotes que identificar com o endereço. Para melhor desempenho do processo o switch terá internamente uma tabela com o endereço *MAC address* de cada máquina conectada em suas portas. Dessa forma, basta comparar o *MAC address* em sua tabela e já poderá identificar qual a porta de destino. Para preencher a tabela, cada vez

que o switch recebe um pacote com um *MAC address* de destino que ele não encontra na tabela, o switch envia para todas as portas a requisição na primeira vez. Assim que a máquina destino responde, ele insere na tabela o *MAC address* do dispositivo que respondeu e em qual porta ele está conectado.

Quando uma máquina precisar enviar uma informação para todas as placas de rede conectadas ao *HUB* ou *Switch*, pode usar um endereço especial com todos os 48 bits ligados. O endereço em hexadecimal fica FF:FF:FF:FF:FF:FF. O endereço indica que todas as máquinas podem aceitá-lo e verificar seu conteúdo. Esse tipo de mecanismo, em que uma informação é enviada para todos, é chamado de *Broadcast*. Veremos algumas aplicações desse tipo de endereço ao verificar o protocolo ARP e DHCP mais adiante.

### 2.3.2 Tipo de protocolo

No campo seguinte aos endereços, temos o tipo de protocolo que está sendo enviado dentro do campo de dados, mas apenas caso o valor seja maior que 0x0600 em hexadecimal (ou 1.536 em decimal). Se o valor for menor ou igual, é interpretado como o tamanho do pacote Ethernet. Os exemplos dos tipos mais utilizados atualmente estão no quadro a seguir.

Quadro 2.2 – Códigos de tipo de protocolo

Código	Descrição
0x0800	Indica que o protocolo inserido no campo de dados do Ethernet é um IPv4. Esse protocolo é um dos mais importantes da arquitetura TCP/IP e será visto com detalhes em outros capítulos.
0x0806	Indica que o protocolo inserido no campo de dados do Ethernet é o ARP. Esse protocolo permite identificar a qual <i>MAC address</i> o IP informado pertence.
0x86DD	Indica que o protocolo inserido no campo de dados do Ethernet é um IPv6, que é a nova geração do protocolo IPv4, e também será visto com detalhes em outros capítulos.
0x8100	Esse código é utilizado para criar uma subdivisão na rede local chamada de VLAN (Virtual LAN). Veremos este tipo de mecanismo mais adiante neste capítulo.

Fonte: Elaborado pelo autor com base em IETF (IETF, 2013).

### 2.3.3 Dados

Nesse espaço, são inseridos os dados transmitidos ao outro equipamento – isso, na verdade, significa que será encapsulado outro protocolo nesse campo. O conceito de encapsulamento é utilizado em diversas situações dentro da área de redes e outras áreas da informática.

Com relação às redes, veremos que todos os protocolos utilizados para comunicação na internet e nas redes locais envolvem algum tipo de inserção de um protocolo no campo de dados do protocolo anterior. Os computadores ficam “descascando” cada camada de protocolo até chegar ao último, em que estão efetivamente os dados que foram encaminhados. Em várias dessas fases é feita uma verificação se o pacote está íntegro, e para isso temos códigos que são acrescentados no pacote para auxiliar a conferência. No caso do Ethernet, o código para isso é o CRC e fica no final do pacote.

### 2.3.4 Código de conferencia (CRC)

Esse código permite que seja feita uma conferência dos bits transmitidos dentro do quadro da Ethernet. Esse campo representa um número que é o resultado de um cálculo matemático envolvendo os bits do quadro Ethernet. Caso algum bit seja alterado durante a transmissão do pacote, o cálculo é refeito no receptor e o resultado vai ser diferente. Nesse caso, o receptor pode descartar o pacote, pois ele não está íntegro.

Além dos campos padrões do Ethernet, foram criadas tecnologias para auxiliar o desenvolvimento de topologias de rede usando os ativos de maneira mais flexível, envolvendo a separação lógica das redes ou facilitando a instalação de ativos. Esses padrões acrescentam mais informações ao pacote Ethernet padrão.

## 2.4 IEEE 802.1Q (VLAN)

Quando estamos configurando uma rede numa empresa, muitas vezes existe a necessidade de separação de parte da rede para isolar alguns dispositivos dos demais. Nesses casos, a colocação de mais um *switch*, que deixa os dois *switches* sem ligação, bastaria para uma rede não “enxergar” a outra. No

entanto, isso implicaria a compra de mais um equipamento, na alteração do cabeamento e no desperdício de portas que teríamos em cada *switch*.

Uma forma de ter o mesmo resultado seria a criação de uma **rede virtual** em que um grupo de máquinas estaria agrupada e separada por uma “**etiqueta**” das outras máquinas do *switch*. As etiquetas numeradas indicariam que uma máquina apenas poderia “conversar” com outra máquina que tenha o **mesmo número** de etiqueta. O termo VLAN vem desse conceito de rede virtual: **Virtual LAN**. E a etiqueta, ou código, é chamada de TAG.

O padrão 802.1Q indica exatamente isto: um código numérico é acrescentado no quadro da Ethernet indicando que pertence a determinado grupo. Esse código pode ser configurado diretamente no *switch* indicando quais portas participam de um mesmo grupo. Assim, todas as máquinas que estejam conectadas nas portas que tenham a mesma TAG podem estabelecer a conexões entre elas. Quando tentarem se comunicar com outras máquinas que estejam em portas configuradas com outro código de TAG não conseguem – como se parte do *switch* fosse “cortado” e fosse interrompida a comunicação com a outra parte.

Caso exista a necessidade de fazer uma comunicação entre as duas VLANs, é feito o mesmo processo de interligação entre redes: com o uso de um roteador as duas redes podem trocar informações entre elas.

Um *switch* já vem configurado com uma VLAN padrão (VLAN 1), e todas as portas inicialmente participam desta VLAN. Uma porta de um *switch* pode participar de mais de uma VLAN. Assim, ao interligar outro *switch*, os pacotes Ethernet que pertençam a diferentes VLANs podem ser transportados para o outro *switch*.

## 2.5 PoE (Power over Ethernet ou energia via cabo Ethernet)

Existem diversos tipos de dispositivos que podem ser atualmente ligados em rede, como câmeras de vigilância, pontos eletrônicos, pontos de acesso sem fio, central de alarmes e equipamentos de automação predial. Quando se inicia a distribuição desses dispositivos na empresa, percebemos que nem

todos os locais apresentam toda a infraestrutura necessária para a ligação do dispositivo. É comum casos em que exista a necessidade de instalar uma câmera de vigilância em um ponto perto de uma entrada onde não há energia elétrica perto.

Normalmente, para cada equipamento a ser instalado na rede existe a necessidade de um ponto de acesso da rede propriamente dito e de um ponto de energia elétrica, o que implica o custo de dois cabeamentos diferentes. Com a adicional preocupação que um cabeamento de energia elétrica normal (127 v ou 220 v) pode criar interferência no cabo de rede se ele não estiver a uma distância segura ou cuidados adicionais na tubulação.

O padrão PoE envolve uma maneira de enviar energia pelos fios do cabo UTP. Seguindo esse padrão, diferentes fornecedores podem utilizar o conceito sem correr o risco de queima do equipamento.

## 2.6 Padrão 802.11 (wireless – sem fio)

Outro padrão em uso nas empresas é a comunicação sem fio 802.11. Como vimos no capítulo anterior, o meio de comunicação sem fio tem algumas vantagens na sua implantação. Com a padronização pelo IEEE das redes wireless, o uso do pacote Ethernet foi mantido, ou seja, podemos aproveitar o conhecimento da sua estrutura na rede sem fio inclusive os formatos de endereço (*MAC address*) discutidos na seção da Ethernet.

O funcionamento desse padrão nos *access point* é semelhante ao do HUB. Todos os equipamentos que estão dentro da área de abrangência recebem o pacote que está sendo transmitido, e a capacidade da rede é compartilhada pela quantidade de equipamento que está conectado à rede.

Como um ambiente pode ficar rapidamente sobrecarregado, e a comunicação pelo ar é aberta, foi criado um identificador para assegurar que cada equipamento esteja se comunicando apenas com os outros que fazem parte do seu grupo. Esse identificador, chamado de SSID (*Service Set Identifier* – ou identificador de um grupo de serviço), permite que o usuário reconheça a string de 32 bytes que identifica a rede que deseja participar, e a comunicação entre as máquinas fique isolada de outros grupos.

O SSID também orienta a troca de chaves de autenticação entre os dois equipamentos. Quando habilitada, essa autenticação assegura que apenas dispositivos com permissão de acessar a comunicação do grupo consigam estabelecer contato com o grupo. Quando não habilitada a autenticação, o SSID apenas indica a rede.

Existe a possibilidade de computadores (ou notebooks) estabelecerem a comunicação diretamente uns com os outros. Dessa forma, a rede se constitui dinamicamente, sem a necessidade de um ativo central intermediando a comunicação dos equipamentos. Esse tipo de rede se denomina “ad hoc” e tem sido objeto de estudos nos últimos anos, ainda mais agora com a possibilidade de diversos aparelhos poderem se conectar a uma rede.

### Saiba mais

A IoT (*Internet of Thing* ou internet das coisas) é um conceito relacionado a como objetos simples do dia a dia estão sendo interligados a rede de computadores. Relógios, TVs, tênis, roupas, sensores dos mais diversos, desde um marca-passos até um chuveiro, são exemplos de novos dispositivos que podem transmitir informações.



## 2.7 Simuladores

Os cenários de redes estão cada vez mais complexos, envolvendo vários protocolos e equipamentos além de formas de conexão diferentes em cada empresa. E nem sempre é possível ter equipamentos à disposição para testar alternativas ou novos procedimentos na rede, pois ter equipamentos de reserva para esses procedimentos tem um custo adicional que a empresa muitas vezes não tem condições de bancar.

Entretanto, a evolução da tecnologia, principalmente com relação à virtualização, tem trazido vantagens também para a simulação de configurações de redes e seus protocolos. Um cenário de uma empresa pode ser testado inicialmente via software e depois que as principais condições forem simula-

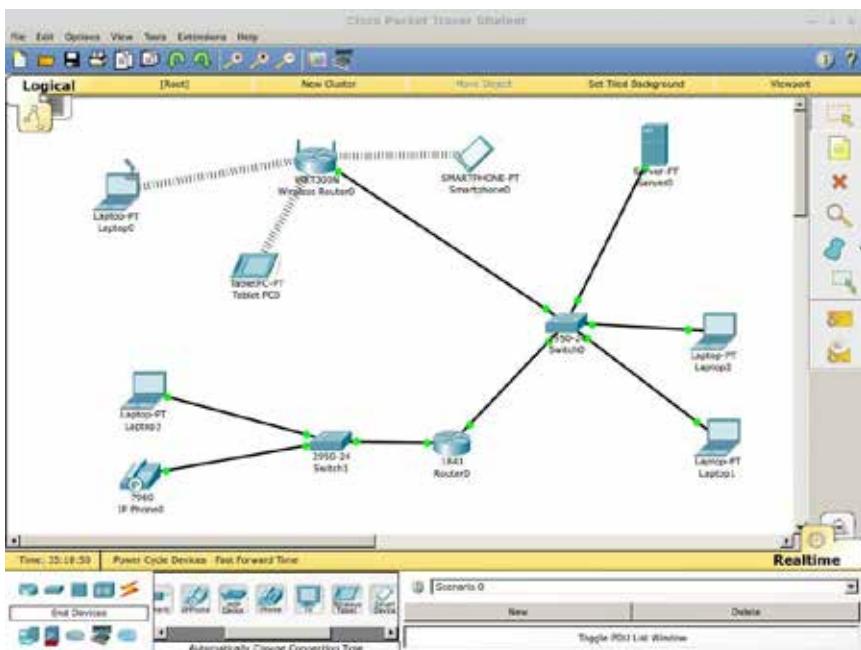
das, a implantação da nova proposta fica mais assertiva e a empresa fica mais segura de realizar o investimento.

Muitos softwares podem ser utilizados para o teste de redes de computadores. Vamos conhecer três deles.

### 2.7.1 Packet Tracer (Cisco)

Esse software é proprietário e foi desenvolvido pela Cisco (uma das principais fabricantes de equipamentos de rede) como ferramenta de apoio para seus alunos nos cursos da Cisco Academy. O Packet Tracer dispõe de objetos que simulam o funcionamento de seus roteadores, switches, access points e também tem objetos com funcionalidades semelhantes a computadores, tablets, telefone IP e servidores para gerar e consumir tráfego de rede (figura 2.4).

Figura 2.4 – Configuração de rede no Packet Tracer



O programa, nas versões mais recentes, permite integrar o diagrama montado com outras redes reais. Podem ser criados testes para alunos reproduzirem durante as práticas.

Os objetos servidores têm serviços de rede que simulam um site, serviço DHCP, sincronização de hora e outros que permitem uma visão dos protocolos utilizados no ambiente.

Apesar de ser simples de utilizar ele tem condições de testar diferentes situações de roteadores e inclusive utilizando os comandos usados para configurar um roteador real, pois reproduz a mesma interface de linha de comando dos roteadores do fabricante.

## 2.7.2 CORE

O software CORE (*Common Open Research Emulator*) foi desenvolvido pela divisão de tecnologia e pesquisa da Boing e atualmente é mantido pelo laboratório de pesquisa Naval dos Estados Unidos (*Naval Research Laboratory – NRL*) como um projeto open source, disponível para Linux e Freebsd (mas pode ser usado em máquinas virtuais no Windows).

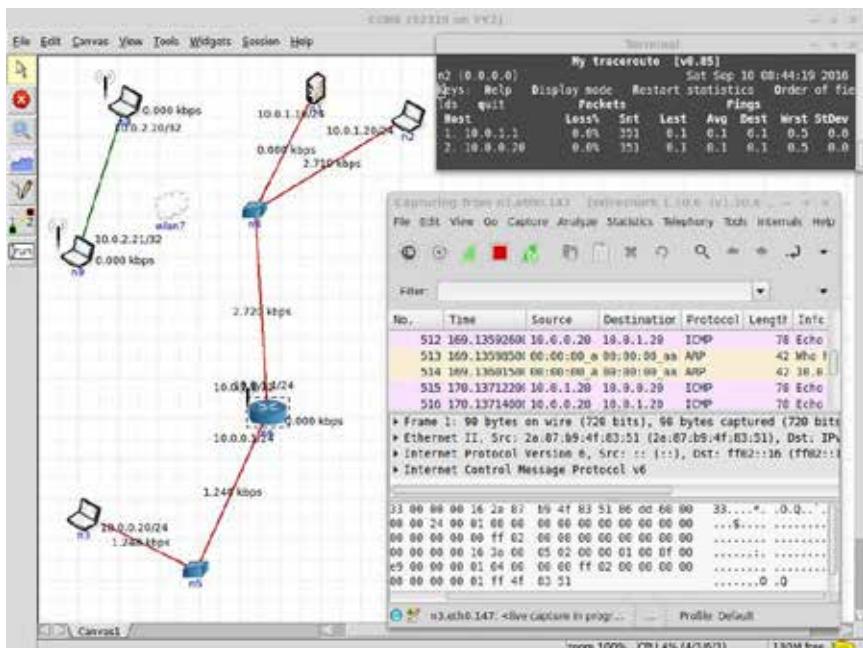
Diferente do Packet Tracer, o sistema CORE usa a virtualização para executar as simulações de rede. Os objetos computador e servidores são ambientes virtuais isolados do sistema operacional em que o CORE está instalado. Podem ser utilizadas as mesmas ferramentas de diagnóstico e teste que seriam usadas num servidor Linux e Freebsd real. O roteador é um sistema de roteamento chamado **Quagga**, um projeto *open source* usado em equipamentos reais para executar as funcionalidades avançadas de roteamento de rede. Com isso, o CORE não depende de sistemas proprietários (como o IOS da Cisco) para realizar o roteamento entre as redes.

Como o CORE executa sistemas do tipo Unix (Linux ou Freebsd) como servidores da rede, programas consagrados de uso na internet como o Apache (servidor web), BIND (servidor DNS) e SQUID (proxy web) – apenas para citar alguns – podem ser usados para simular uma aplicação real em uso e, com isso, acompanhar como a rede se comporta.

Nos objetos inseridos na rede é possível executar alguns analisadores de rede como o Wireshark e Tcpdump (vistos na seção sobre Utilitários

Farejadores neste capítulo) e que mostram os pacotes transmitidos e recebidos pelos objetos.

Figura 2.5 – Configuração de rede no CORE usando Wireshark e MTR



O sistema CORE pode também ser interligado a redes físicas reais e interagir com elas pela interface de rede do sistema hospedeiro, assim como se comunicar com outras configurações CORE.

### Saiba mais

O programa CORE é utilizado pelo NIC.br (núcleo do Comitê Gestor da Internet no Brasil) para testes e divulgação do protocolo IPv6 no Brasil. No site [ipv6.br](http://ipv6.br), existem livros, apostilas e experiências de laboratório usando o CORE para testes com ambientes de redes com o novo protocolo.

Caso seja necessário, podem ser criados scripts em Python para automatizar algum teste de rede mais avançado.

Em cenários muito complexos, envolvendo diversos dispositivos, pode-se executar vários trechos do projeto da rede em máquinas diferentes com o CORE instalado. Ao mesmo tempo, pode-se visualizar toda a rede num único diagrama. Esse diagrama com a visão de todos os pontos pode estar numa máquina separada das máquinas destinadas a execução das emulações.

### 2.7.3 GNS3

O GNS3 é um emulador de redes que utiliza a emulação dos sistemas dos roteadores CISCO como principal opção, mas também permite a utilização de outros tipos de roteadores. Como o CORE, ele utiliza a virtualização de sistemas reais para a criação do ambiente de rede que será testado, com isso é possível analisar como um sistema se comportará em determinadas situações.

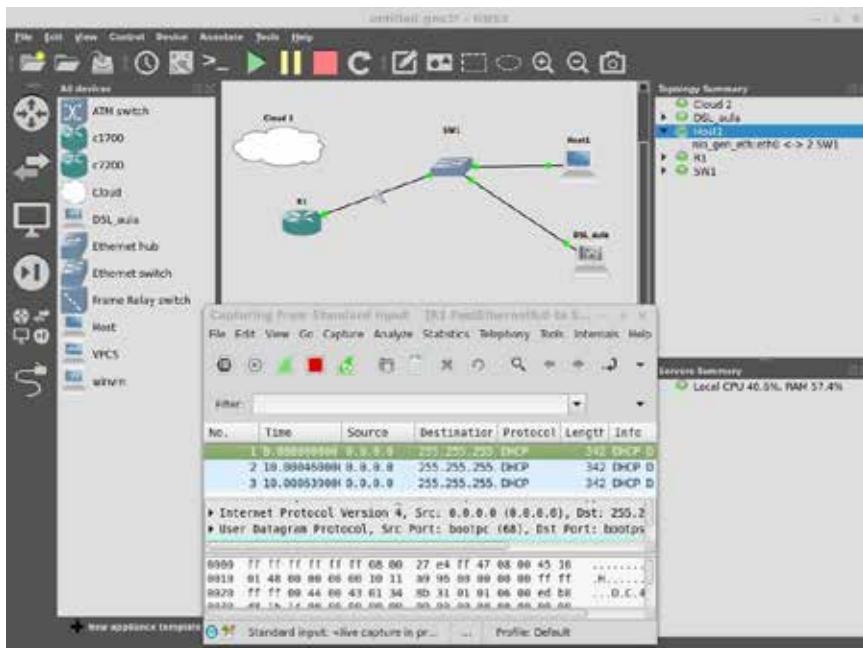
O GNS3 iniciou como uma interface gráfica para o pacote “Dynamips”, um emulador de processador MIPS usado nos roteadores CISCO e responsável por executar o sistema operacional desses equipamentos (IOS – *Internetwork Operating System*) de maneira a simular o funcionamento de redes que teriam roteadores CISCO como estrutura base. Atualmente, além do Dynamips e a virtualização de sistemas operacionais, pode utilizar também serviços de roteamento *open source* como o Quagga, semelhante ao emulador CORE.

Como o IOS é um sistema operacional proprietário, ele não é instalado juntamente com o GNS3, e é preciso conseguir junto a CISCO uma imagem do sistema. Com essa imagem, o Dynamips inicia o sistema e habilita o roteador para ser configurado via linha de comando.

O GNS3 integra máquinas virtuais com Linux e Freebsd usando como receptores ou transmissores de informação na rede. Essas conexões podem ser inspecionadas com o Wireshark e Tcpdump.

## Rede de Computadores

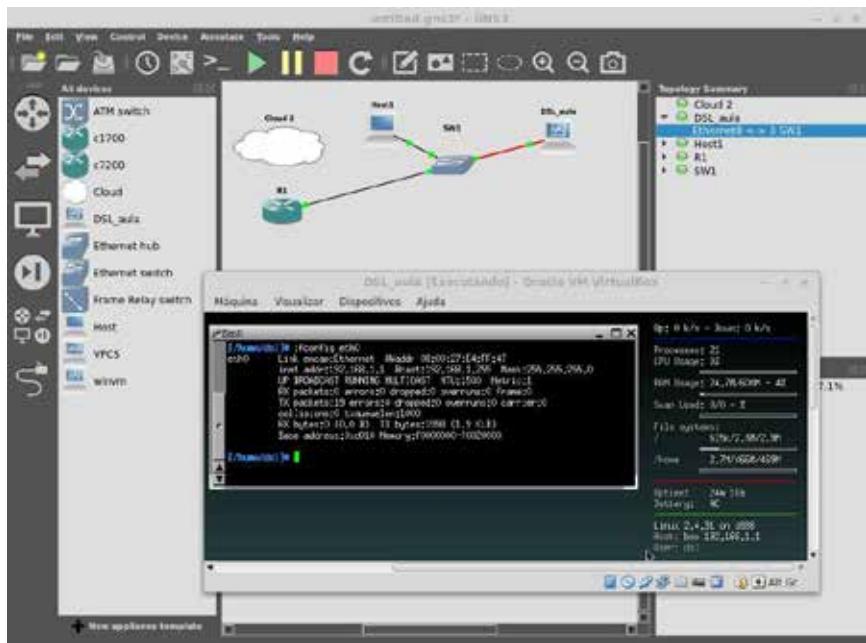
Figura 2.6 – Configuração de rede no GNS3 usando Wireshark



Na lista de equipamentos a serem inseridos no diagrama da rede, é possível fazer um rastreamento nas máquinas virtuais disponíveis (salvas pelo VirtualBox no computador em que está sendo executado o GNS3), e essas máquinas são trazidas como opção na lista de dispositivos (*devices*). Ao executar o ambiente projetado, a máquina virtual é inicializada, e a interface de rede é interligada no diagrama, o que faz com que as informações enviadas ou recebidas passem para o sistema operacional sendo executado.

Na figura 2.7, há um sistema Linux (DSL\_aula) sendo executado pelo GNS3, e é possível verificar no quadro “*Topology Summary*”, no canto superior direito, a integração da placa Ethernet do equipamento com a porta 3 do Switch (Ethernet - 3 SW1).

Figura 2.7 – Execução de sistema virtual integrado no diagrama de rede



Alguns dispositivos podem ser instalados usando a plataforma *Docker*, facilitando a disponibilidade de novas funcionalidades dentro da rede sendo testada.

## 2.8 Utilitários farejadores (sniffers)

Existem utilitários que permitem visualizar o que está sendo transmitido entre uma máquina e outra e, com isso, podemos avaliar um problema ou o funcionamento de um novo serviço na rede. Conhecidos como “farejadores” (*sniffers*), esses programas capturam todos os pacotes percebidos pela placa de rede, mesmo que não tenham o *MAC address* da placa de rede como destino.

Para isso, é ligado o modo “*promíscuo*” da placa de rede, e ela passa a receber todos os pacotes enviados ao meio de transporte que ela tem acesso (seja por cabo ou pelo ar).

Dois utilitários serão vistos neste capítulo: Tcpdump e Wireshark.

### 2.8.1 Tcpdump

Esse utilitário foi um dos primeiros a se aproveitar da característica dos HUBs de replicar o pacote para todas as portas e das placas de rede que apresentam um “**modo promíscuo**”. Ele permite utilizar uma série de filtros (endereço, porta, interface, tipo de protocolo) e, com isso, selecionar a sequência de pacotes que precisa ser analisada.

Nos capítulos seguintes, veremos outros protocolos que contêm informações diferentes em seus pacotes e que também podem ser utilizados nos filtros do tcpdump para a seleção.

Como exemplo, podemos ver a seguir a sequência de captura de uma comunicação entre duas máquinas. No caso, foi enviado um pacote para testar a comunicação em uma interface de teste (*loopback*) chamada **localhost**. Essa interface simula a resposta de uma placa de rede a um comando de teste chamado **ping**. O comando **ping** será muito útil quando tivermos no capítulo sobre TCP/IP vendo a camada de rede.

---

Apesar de extremamente simples o comando ping é utilizado como uma das primeiras peças no arsenal de testes de rede. Ele envia um pacote de teste para o endereço de outra máquina e, quando a máquina o recebe, envia um pacote de resposta. Com isso, podemos entender que todo o trajeto da rede foi percorrido e as duas máquinas estão aptas a enviar informações pela rede. O nome do comando foi baseado na lógica dos sonares que enviavam um pulso sonoro (ping) para saber se existe algum obstáculo no caminho.

---

No quadro 2.3, temos o resultado do comando em linhas numeradas para facilitar a explicação.

A linha (1) mostra o comando executado com a opção de usar endereços numéricos (-n), mostrar informações do pacote Ethernet (-e) e usar a interface de teste localhost.

Quadro 2.3 – Resultado da captura via Tcpdump (ping)

```

1. tcpdump -n -e localhost
2.
3. tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
   listening on lo, link-type EN10MB (Ethernet), capture size 65535 bytes
4.
5. 21:54:18.903702 00:00:00:00:00:00 > 00:00:00:00:00:00, ethertype IPv4 (0x0800), length 98:
   127.0.0.1 > 127.0.0.1: ICMP echo request, id 27858, seq 1, length 64
6.
7. 21:54:18.903732 00:00:00:00:00:00 > 00:00:00:00:00:00, ethertype IPv4 (0x0800), length 98:
   127.0.0.1 > 127.0.0.1: ICMP echo reply, id 27858, seq 1, length 64

```

Fonte: Elaborado pelo autor.

Na linha 3, temos apenas o cabeçalho do programa. Na linha 5, temos o primeiro pacote enviado para o teste. Em negrito estão os endereços *MAC address* da placa de teste (**00:00:00:00:00:00**), que no caso estão todos zerados, e o tipo de protocolo que foi encapsulado pelo protocolo Ethernet (**0x0800**), que o quadro indica ser um IPv4. Nessa linha também temos o caractere “>” indicando de quem e para quem é a conexão. Nesse caso, acaba não sendo útil, por estarmos usando a interface de teste, mas mais abaixo teremos uma conexão entre duas máquinas com endereços diferentes. Na linha 7, temos o pacote de resposta do teste.

No próximo quadro (2.4), temos outra sequência capturada pelo tcpdump. Na primeira linha, temos o comando executado. Dessa vez foi solicitado que a placa de rede wireless fosse o alvo da captura (**-i wlan0**). Na linha 6, temos o envio de um pacote da máquina (**80:29:94:e1:ee:0f**) para todas as máquinas da rede usando o endereço especial de broadcast (**ff:ff:ff:ff:ff:ff**). O tipo de pacote é ARP (**0x0806**) que veremos com detalhe em outro capítulo. Novamente o caractere “>” indica o sentido da comunicação.

#### Quadro 2.4 – Resultado da captura via Tcpdump (ARP)

```
1. tcpdump -i wlan0 -n -e
2.
3. tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
4. listening on wlan0, link-type EN10MB (Ethernet), capture size 65535 bytes
5.
6. 22:12:28.527873 80:29:94:e1:ee:0f > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Request
  who-has 192.168.0.13 tell 192.168.0.1, length 46
7.
8. 22:12:28.527888 0c:d2:92:92:94:b5 > 80:29:94:e1:ee:0f, ethertype ARP (0x0806), length 42:
  Reply 192.168.0.13 is-at 0c:d2:92:92:94:b5, length 28
```

Fonte: Elaborado pelo autor.

Na linha 8, temos a resposta de uma das máquinas da rede (**0c:d2:92:92:94:b5**) para a máquina que “lançou” a pergunta para todos (**80:29:94:e1:ee:0f**).

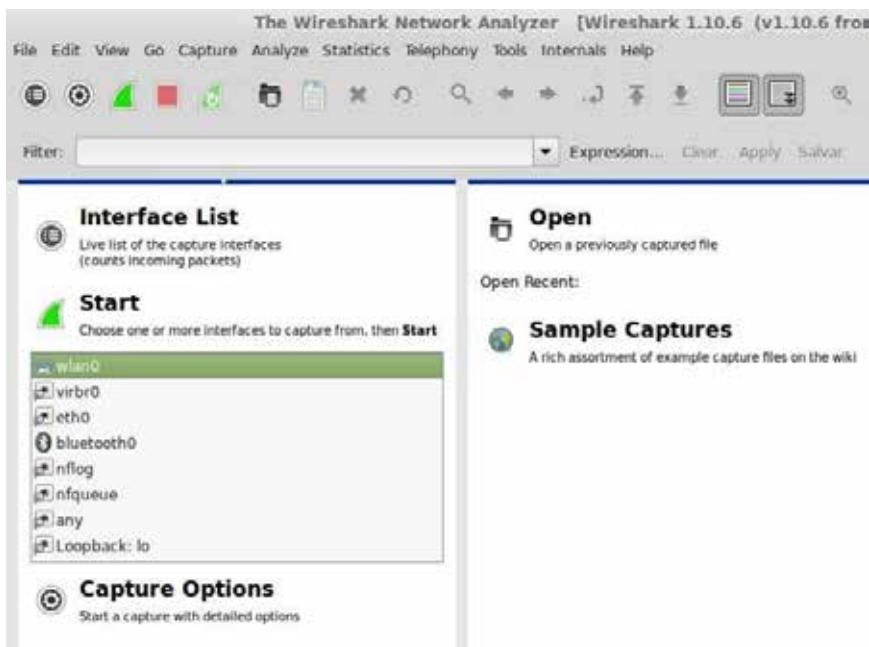
### 2.8.2 Wireshark

O utilitário WireShark tem o mesmo conceito do tcpdump, mas utiliza uma interface gráfica que auxilia na análise do conteúdo dos pacotes e dos níveis de encapsulamento utilizados durante a transmissão da informação.

Ele captura a comunicação pela interface de rede selecionada e depois apresenta os pacotes capturados separando cada protocolo e possibilitando abrir cada um e verificar todas as informações, tanto dos cabeçalhos do pacote quanto dos dados enviados.

Na figura seguinte, temos a tela principal do software com a lista de interfaces (wlan0, virbr0, eth0, bluetooth0, nflog, nfqueue, any, loopback:lo) que podem ser usadas para a captura do tráfego de rede. O Wireshark também pode ser utilizado para analisar uma captura feita pelo tcpdump. Nesse caso, utiliza-se o tcpdump para fazer a captura e salvar as informações em um arquivo. Depois, na fase de análise das informações, podemos abrir o arquivo dentro do Wireshark.

Figura 2.8 – Tela inicial do Wireshark com a lista de interfaces



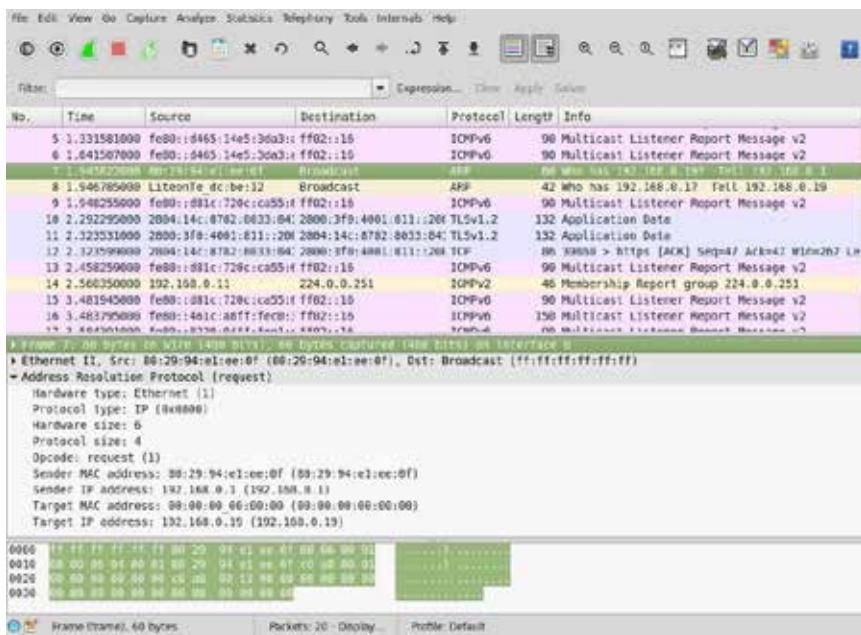
Na figura 2.9, temos uma captura semelhante à sequência feita pelo tcpdump (ARP). A tela do wireshark é dividida em três partes: uma parte mostra quem está se conectando com quem e com qual protocolo (semelhante ao resultado do tcpdump), na segunda parte temos todos os protocolos envolvidos na comunicação e na última temos os dados transferidos.

Nesse exemplo, na primeira parte da tela a linha marcada mostra um dos registros da comunicação. As informações estão separadas por colunas e temos principalmente o tempo entre um registro e outro, a coluna de origem (*source*) da transmissão e o destino (*destination*). Nesse registro a máquina de origem (*MAC address 80:29:94:e1:ee:0f*) está enviando para todos (*Broadcast*).

(*cast*). Na próxima coluna, é mostrado o protocolo que está sendo encapsulado (*protocol*) e na coluna seguinte, o tamanho do pacote (*length* de 60 bytes).

Para facilitar a leitura das informações, o Wireshark permite a colorização dos registros conforme o protocolo, filtros para selecionar apenas a comunicação a ser analisada e a ordenação das informações por algum tipo de coluna. Por esse motivo, o Wireshark é muito utilizado também para analisar tráfego capturado pelo tcpdump visto acima. O tcpdump executa a captura e salva num arquivo e o Wireshark já possui compatibilidade com o tipo de arquivo para abrir as informações e facilitar a análise.

Figura 2.9 – Tela com as informações capturadas pelo Wireshark



Nas outras partes da tela, temos como verificar as informações de cada protocolo (na tela temos o protocolo ARP em detalhes) e, na última parte da tela, temos o conteúdo da parte de dados. Como o protocolo ARP não tem nenhuma informação, são mostrados pontos e outros caracteres sem sentido.

Se estivéssemos acessando um site, nessa parte seriam mostrados os caracteres que compõe a página html no lado direito e o código hexadecimal correspondente no lado esquerdo.

O Wireshark ainda permite relacionar uma série de estatísticas dos dados dos pacotes capturados: ocorrência de um protocolo, quantidade de bytes transmitidos ou o fluxo entre dois pontos. Alguns desses dados podem ser transformados em gráficos pelo próprio programa.

Os dados de uma sessão de captura podem ser segmentados em tipos de protocolos ou características do endereço Ethernet e montado um gráfico com o resultado (figura 2.10).

Figura 2.10 – Janela de gráficos de bytes entrantes e saítes do Wireshark



Nesse gráfico, foram separados os pacotes dos protocolos http, dns e ssl, e para cada um foi atribuída uma linha do gráfico geral. Pode-se indicar o estilo a ser utilizado no desenho do ponto: linha, preenchido ou pontilhado, e cada item fica com uma cor no gráfico geral. No caso anterior, não foi indi-

cado nenhum filtro para a cor verde, significando que será usado o total de bytes capturado nesta faixa.

### 2.8.3 Utilitário de Linha de comando

Alguns utilitários de linha de comando podem ser usados para identificar informações relacionadas às interfaces que utilizam o protocolo Ethernet. Utilizaremos o sistema Linux como base para os testes de linha de comando.

Primeiramente, temos o *lshw* (lista o hardware) e, caso não esteja instalado como padrão as distribuições Linux, permite baixar o utilitário e instalar usando comandos de instalação. Nas distribuições baseadas em Debian (como a Mint utilizada nos exemplos) permite que seja instalado usando “*apt-get install lshw*”. A seguir, temos o resultado da execução do comando “***lshw -C network***”. Ele com a opção para mostrar apenas as informações relacionadas à rede (-C network) mostra informações como fabricante e capacidade da placa. As principais informações estão em negrito no quadro 2.5.

Quadro 2.5 – Resultado do comando *lshw* contendo interface com fio (eth0) e sem fio (wlan0)

```
*-network
      descrição: Ethernet interface
      produto: QCA8172 Fast Ethernet
      fabricante: Qualcomm Atheros
      ID físico: 0
      informações do barramento: pci@0000:01:00.0
      nome lógico: eth0
      versão: 10
      serial: 64:1c:67:63:72:8c
      tamanho: 100Mbit/s
      capacidade: 100Mbit/s
```

```
largura: 64 bits
clock: 33MHz
capacidades: pm pciexpress msi msix bus_master cap_list
list ethernet physical tp 10bt 10bt-fd 100bt 100bt-fd
autonegotiation
configuração:      autonegotiation=on      broadcast=yes
driver=alx duplex=full ip=192.168.50.112 latency=0
link=yes multicast=yes port=twisted pair
speed=100Mbit/s
recursos: irq:46 memória:e0500000-e053ffff porta de
E/S:2000(tamanho=128)
*-network
    descrição: Interface sem fio
    produto: Centrino Wireless-N 135
    fabricante: Intel Corporation
    ID físico: 0
    informações do barramento: pci@0000:02:00.0
    nome lógico: wlan0
    versão: c4
    serial: 0c:d2:92:92:94:b5
    largura: 64 bits
    clock: 33MHz
    capacidades: pm msi pciexpress bus_master cap_list
list ethernet physical wireless
    configuração:      broadcast=yes      driver=iwlwifi
driverversion=3.13.0-24-generic firmware=18.168.6.1
ip=192.168.0.13 latency=0 link=yes multicast=yes
wireless=IEEE 802.11bgn
    recursos: irq:44 memória:e0400000-e0401fff
```

Fonte: Elaborado pelo autor.

No quadro 2.6, estão indicados também o *MAC address* da interface, mas sob o nome “**serial**”.

Outro utilitário usado para verificar mas também **alterar** características da placa de rede é o *Ethtool*. Esse utilitário é usado principalmente para redes com fio e, do mesmo modo que o comando *lshw* anterior, pode ser instalado usando o comando “*apt-get install ethtool*”.

Quadro 2.6 – Resultado do comando “ethtool eth0”

```
Settings for eth0:  
Supported ports: [ TP ]  
Supported link modes: 10baseT/Half 10baseT/Full  
                      100baseT/Half 100baseT/Full  
Supported pause frame use: Symmetric Receive-only  
Supports auto-negotiation: Yes  
Advertised link modes: 10baseT/Half 10baseT/Full  
                      100baseT/Half 100baseT/Full  
                      1000baseT/Full  
Advertised pause frame use: Symmetric  
Advertised auto-negotiation: Yes  
Speed: 100Mb/s  
Duplex: Full  
Port: Twisted Pair  
PHYAD: 0  
Transceiver: internal  
Auto-negotiation: on  
MDI-X: Unknown  
Current message level: 0x000060e4 (24804)  
                      link ifup rx_err tx_err hw wol  
Link detected: yes
```

Fonte: Elaborado pelo autor.

No caso do *ethtool*, é possível alterar alguma característica desde que o parâmetro seja suportado pelo driver ou pela placa de rede. Por exemplo, o comando “*ethtool -s eth0 speed 10*” altera a velocidade da placa de rede de 100 Mbits (mostrado na tabela) para 10 Mbits.

Para interface sem fio, há o “*iwlist*” que pode listar as redes sem fio visíveis pela placa, assim como as características de cada uma. Esse utilitário deve ser instalado usando “*apt-get install wireless-tools*”, que reúne este e outros comandos para uso na interface sem fio.

Quadro 2.7 – Resultado parcial do comando “*iwlist wlan0 scan*”

```
Cell 03 - Address: B0:48:7A:C7:05:FE

Channel:3
Frequency:2.422 GHz (Channel 3)
Quality=70/70 Signal level=-19 dBm

Encryption key:on
ESSID:"YANKO"
Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 6 Mb/s
         9 Mb/s; 12 Mb/s; 18 Mb/s
Bit Rates:24 Mb/s; 36 Mb/s; 48 Mb/s; 54 Mb/s
Mode:Master
Extra:tsf=00000000d561da72
Extra: Last beacon: 8ms ago
IE: WPA Version 1
    Group Cipher : TKIP
    Pairwise Ciphers (2) : CCMP TKIP
    Authentication Suites (1) : PSK
```

Fonte: Elaborado pelo autor.

No quadro 2.7, entre as redes que a placa sem fio encontrou, existem as informações de uma delas (*Cell 03*) que estão sendo propagadas via o equi-

pamento com o endereço: **B0:48:7A:C7:05:FE**. Essas e outras informações principais foram deixadas em negrito. Nelas, é possível verificar características como o nome da rede “**ESSID**”, se a rede tem criptografia para autenticação “**Encryption key:on**” e o tipo de criptografia “**IE: WPA Version 1**”, assim como as velocidades que a rede pode aceitar em “**Bit Rates**”.

## Síntese

Vimos neste capítulo como funciona o protocolo Ethernet, como é estruturado e como podem ser identificados os equipamentos na rede com base em seus endereços físicos (*MAC address*). Pudemos verificar, dentro dos padrões relacionados ao protocolo, variações e extensões que permitem isolar partes da rede ou fazer a transmissão do pacote sem o uso de cabos.

Para analisar as diversas configurações dos dispositivos e proporcionar o teste prático dos conceitos e condições de redes de computadores corporativas ou residenciais que veremos ao longo destes capítulos, foi introduzido o funcionamento de alguns dos mais conhecidos simuladores de redes. Esses simuladores disponibilizados podem criar redes complexas com vários tipos de dados sendo transmitidos, e depois, pacotes, fluxos e características podem ser detalhados e analisados.

Por último, dois utilitários de captura e inspeção de tráfego foram apresentados. Esses utilitários podem ser usados isoladamente ou em conjunto com os simuladores.

## Atividades

1. O pacote Ethernet tem um formato em que os endereços de origem e destino ficam registrados, permitem encontrar a máquina que precisa receber a informação, assim como identificar a origem da transmissão. Marque a opção correta sobre o endereçamento Ethernet.
  - (A) O endereço da Ethernet tem 32 bits e é separado por quatro grupos de números em decimal. Também é chamado de *MAC address*.

- (B) O endereço da Ethernet tem 3 bytes usados para identificar a empresa que fabricou a placa de rede. Os números são formatados em seis números em hexadecimal separado por “:”.
- (C) Os dois endereços da Ethernet têm tamanho variável e pode chegar a 48 bits. É chamado de *MAC address* e usa os três primeiros bytes para enviar uma mensagem a todos os equipamentos (*broadcast*).
2. Uma empresa dispõe de um switch de 48 portas e tem 20 computadores ligados em rede cabeada. Foram comprados mais 5 computadores para criar um pequeno laboratório para seus treinamentos com representantes, mas a empresa não quer que o laboratório acesse o restante da rede, em que está o sistema administrativo da empresa e demais máquinas e impressoras. Qual tecnologia pode ser usada para isolar este laboratório sem a compra de mais equipamentos.
3. Qual o objetivo do SSID numa rede sem fio?
4. Considerando o texto sobre simuladores, identifique vantagens do uso desse tipo de software para o profissional de redes de uma empresa.



# 3

## Modelo OSI e Arquitetura TCP/IP

NESTE CAPÍTULO, ABORDAREMOS o modelo internacional de referência para construção de redes (OSI – *Open Systems Interconnection*) e a estrutura do conjunto de protocolos que domina as redes locais e de longa distância atualmente, chamado de TCP/IP. Apesar de citada em alguns locais como um protocolo único, a arquitetura TCP/IP envolve uma série de protocolos agrupados em camadas. Veremos como foi criado o conceito por trás dela e sua forma de utilização.

ENTENDEREMOS COMO SÃO documentadas as evoluções e as novas funcionalidade do protocolo e como podem ser conferidos os documentos na internet, assim como os organismos internacionais responsáveis pela guarda e pela evolução das tecnologias.

Desde seu nascimento, na década de 1970, o TCP/IP tem recebido bastante atenção, superando as expectativas quanto a seu uso nas redes de computadores. O conceito original estava relacionado à segurança militar, dentro de uma agência de pesquisa avançada (Darpa – Defense Advanced Research Projects Agency) do Departamento de Defesa dos Estados Unidos. Conforme registrou a Darpa (2015), projetos anteriores de pesquisas, como *Project Lincoln* (desenvolvido no início da década de 1950) e *Project MAC*, do Instituto de Tecnologia de Massachusetts – MIT (em meados da década de 1960), contribuíram com conceitos e ideias para a criação da arquitetura TCP/IP.

O primeiro implantou uma rede de comunicação entre 23 estações de radares na força aérea com interligação de computadores do país por meio de cabos telefônicos; o segundo desenvolveu uma comunidade virtual e funcionalidades para comunicação entre os usuários, como e-mail, murais de avisos e softwares voltados ao uso on-line em terminais de servidores espalhados pelo campus da Universidade.

### 3.1 OSI (Open Systems Interconnection)

Quando há um grande problema a ser trabalhado, as técnicas de análise recomendam dividi-lo em questões menores para facilitar a identificação de uma solução ou alternativa. Em várias situações na informática, temos a necessidade de compartmentalizar as informações e as ações envolvidas, criando uma interface entre os grupos para que se perceba a situação completa. Isso acontece na análise de sistemas, na programação e também em redes.

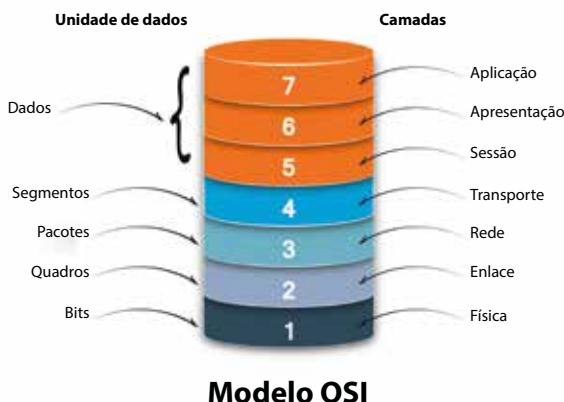
Com o estudo do funcionamento das redes de computadores, foi criado um modelo no qual são separadas as funções principais em camadas (*layers*, em inglês). Essas camadas servem para tratar e organizar a informação que será recebida pelo sistema na outra ponta da comunicação. Como exemplo, temos a situação de correção de erros, sequenciamento, conversão e encaminhamentos exigida pela transmissão de dados em um meio. Com a separação, o desenvolvimento do software que processará cada camada fica mais simples, podendo se concentrar em um grupo menor de tarefas, ficando o algoritmo também menor.

A organização internacional responsável por elaborar um modelo que pudesse ser seguido como padrão para o desenvolvimento de uma estrutura de comunicação foi a ISO (International Organization for Standardization), que estabeleceu sete camadas de funções que se inter-relacionam e podem ser usadas para estabelecer conexão e transmissão de dados entre dois equipamentos, chamadas OSI.

Ao implementar um protocolo, as funções da OSI acabam sendo usadas como referência, mas não necessariamente são implementadas todas as camadas planejadas. Um exemplo é o conjunto TCP/IP, que, apesar de ser mais antigo que o modelo OSI, já utilizava o mesmo conceito e foi segmentado em apenas quatro camadas.

Na figura 3.1, temos uma representação gráfica das camadas com a informação adicional do tipo de representação dos dados usados em cada parte.

Figura 3.1 – Diagrama do modelo OSI



Fonte: Shutterstock.com/Luna2631.

Temos de um lado as sete camadas e do outro lado o formato dos dados utilizados na transmissão. Na camada física, temos os bits, que são agrupados em quadros (*frames*) na camada de enlace e que, ao passar para a camada de rede, são organizados em pacotes (*packets*). Um fluxo de pacotes é visto como um segmento que recebeu os dados (*data*) do programa usado para a conexão.

Para cada camada, foram separadas algumas funções, detalhadas no quadro 3.1.

Quadro 3.1 – Descrição das camadas OSI

Número	Camada	Descrição
1	Física	Relacionada ao meio de transmissão, envolvendo, por exemplo, questões elétricas que serão usadas para identificar os bits 0 e 1, assim como as questões relacionadas a conectores, distâncias e tempos da transmissão e da recepção dos bits.
2	Enlace	Essa camada pretende assegurar que os bits não sofram modificações indevidas (erros) e deixar a sequência de bits pronta para a interpretação da próxima camada. São feitas as separações dos trechos de bits com a inclusão de bits para identificar alteração e também são incluídos os endereços físicos ( <i>MAC address</i> ).
3	Rede	A camada de rede é responsável pelo encaminhamento do pacote desde a origem até o destino, mesmo que envolva pontos intermediários no processo.
4	Transporte	Essa camada cria um padrão para estabelecer a conexão direta entre a origem e o destino. Para isso, é usado um alinhamento entre os dois pontos, estabelecendo as características da transmissão.
5	Sessão	Nessa camada estariam os controles de manutenção e sincronia da transmissão, mantendo os interlocutores com o canal aberto para a comunicação.

Número	Camada	Descrição
6	Apresentação	É uma camada mais relacionada aos dados e à estrutura usada pelos sistemas operacionais dos dispositivos na formatação dos dados.
7	Aplicação	Nesse último nível estariam os protocolos envolvidos diretamente com o usuário. Seria a primeira estruturação da comunicação, feita pelos softwares aplicativos. Normalmente relacionado a compartilhamento de recursos, transferência de arquivos e mensagens.

Fonte: Elaborado pelo autor; OSI (2016); Tannenbaum; Wetherall (2004).

Cada camada trata os dados deixando-os preparados para a camada seguinte. Dessa forma, uma camada “confia” no trabalho da camada anterior e parte do pressuposto de que a informação já está preparada para ser processada.

Na lógica proposta pelo modelo, um programa aplicativo precisa enviar uma informação para outra máquina. Ele estrutura a informação e passa para a camada de baixo, tratada por cada camada e enviada para a camada logo abaixo. Ao chegar na camada física, a informação que já está preparada em um quadro de bits é convertida para a sinalização elétrica correspondente e enviada pela placa de rede ao meio de transmissão.

Do outro lado (receptor), a placa de rede, ao receber a sequência que detecta como sendo para seu endereço, separa os pulsos elétricos, transforma em bits, verifica os erros, agrupa, processa e vai enviando para a camada superior até chegar ao software aplicativo que está associado à camada de aplicação. O programa então acessa a informação e revela ao usuário.

Esse tipo de estruturação permite que o desenvolvimento dos protocolos seja segmentado. Cada protocolo pode se concentrar apenas nas funções a que foi designado. Simplificando a codificação e permitindo que, se um protocolo de uma camada intermediária sofrer uma alteração (seja uma evolução, seja uma substituição), o impacto para as outras camadas seja o mínimo possível.

## 3.2 Domínios

Enquanto bits, bytes, endereços hexadecimais ou números em outras bases são formas em que o computador está preparado para trabalhar, as pessoas, ao tentarem estabelecer uma comunicação, estão mais preparadas para criar nomes e endereços textuais. Nesse aspecto, foram criadas formas de uma pessoa, utilizando uma sentença ou um nome, encontrar um servidor, uma estação e até um dispositivo qualquer e com isso acessar as informações de que precisa.

Mas é necessário certo cuidado ao utilizar nomes. Um nome deve ser único na rede. Da mesma maneira que os endereços numéricos, não podemos ter um mesmo nome direcionado a mais de um servidor. No tocante a uma interligação entre redes de diferentes empresas, não podemos ter um nome conflitando com o nome do servidor em outra empresa. Nesses casos, é possível estabelecer uma hierarquia de nomes, permitindo que sejam agrupados os dispositivos e evitando os conflitos. Quando tratados de maneira hierarquizada, esses nomes são chamados de domínios (*domain*) e são uma importante parte da presença de uma empresa na internet.

### 3.2.1 ICANN

Para certificar que esses nomes possam ser utilizados entre redes, pessoas e empresas sem que haja um conflito, foi definido que uma associação internacional cuidasse dessa situação. A ICANN (The Internet Corporation for Assigned Names and Numbers ou, em tradução livre, Corporação para Delegação de Nomes e Números na Internet) foi criada para desenvolver as funções de gerenciar a distribuição de sequências de números de endereços. Esse tipo de operação é chamado de IANA (*Internet Assigned Number Authority* ou Autoridade de Delegação de Números da Internet).

#### Saiba mais

No início da internet, ainda dentro do Projeto Darpa do Departamento de Defesa Americano, todo o trabalho de distribuição e gerenciamento de endereços era feito por apenas uma pessoa (Jon Postel). Com o crescimento da interligação das redes, foi necessário passar esse trabalho para uma organização independente.

Mesmo para uma organização internacional, à medida que mais países se interessavam em conectar suas redes à arquitetura que estava se formando, crescia a dificuldade de entender as necessidades de cada região. Assim, foi necessário também que houvesse uma administração regional que pudesse avaliar as necessidades de partes específicas do Planeta.

Foram criadas cinco regionais responsáveis por grandes áreas geográficas pertencentes a vários países próximos e com características semelhantes (figura 3.2).

Figura 3.2 – Mapa das registradoras regionais de Internet



Fonte: NRO (2016).

Essas regionais são chamadas de RIR (*Regional Internet Registries* ou Registradoras Regionais de Internet), responsáveis por distribuir os grupos de endereços numéricos para os países da região e organizar a tradução dos nomes para os endereços numéricos.

Considerando a escala global que se tornou a interligação das redes de computadores, o risco de um nome entrar em conflito com outro cresceu muito. Para que se eliminasse o risco de um mesmo nome ser utilizado entre países, foi designado mais um nível de segmentação dos domínios. Para isso, adotou-se a utilização de duas letras padronizadas pela ISO para a indicação de cada país. Por exemplo, BR indica Brasil, DE representa Alemanha, AR faz referência à Argentina. Cada país é responsável por identificar qual grupo ou nome está cadastrado sob sua sigla.

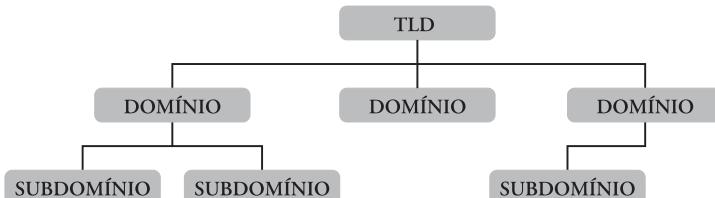
## Saiba mais

É possível verificar a lista de todos os países com suas abreviações usando a base de dados do padrão ISO 3166 disponível on-line (<<https://www.iso.org/obp/ui/#search>>). Nela estão a descrição e as duas letras utilizadas para identificar o país, além de informações sobre o idioma e as subdivisões (estados, distritos ou regiões) com suas respectivas abreviações padronizadas.

Para padronizar os nomes e identificar globalmente a localização de um servidor, foi criada uma estrutura hierárquica para dividir o endereço em partes menores até chegar ao servidor procurado. Cada termo seria separado por um ponto ( . ) para indicar uma subdivisão do domínio.

Figura 3.3 – Hierarquia dos domínios

SUBDOMÍNIO . DOMÍNIO . TLD



Fonte: Elaborada pelo autor.

Para chegar a um servidor, podemos utilizar uma sequência de palavras (da direita para a esquerda) que devem ser seguidas até o termo que indica o servidor (o nome mais à esquerda). Na figura 3.3, temos como sequência primeiro a análise do TLD, depois do DOMÍNIO e por último do SUBDOMÍNIO.

Uma série de siglas está associada à forma de determinar qual conjunto de termos indicará o endereço da rede ou do servidor. No Quadro 3.2, temos as siglas e o significado de cada uma delas. Para cada situação foram estabelecidos padrões de como cada domínio e subdomínio podem ser formatados sob os cuidados do ICANN.

Quadro 3.2 – Siglas associadas aos domínios

Sigla	Descrição
<b>TLD (Top Level Domain)</b>	Indica o termo final do endereço. Apesar de ser o último nome associado ao endereço, é o primeiro termo a ser usado para indicar a qual região ou segmento de rede está ligado. Um TLD pode estar associado a um gTLD ou ccTLD.
<b>gTLD (Generic Top Level Domains)</b>	Nesse caso, o TLD está associado a um termo genérico usado para agrupar os que possuem uma mesma característica. Exemplo de nomes genéricos são .com, .museum, .org. No início da criação da internet, foram definidos oito nomes que poderiam ser usados como TLD: .com, .edu, .gov, .int, .mil, .net, .org e .arpa. Com a evolução da grande rede, foram criados mais alguns nomes genéricos para serem utilizados na segmentação dos domínios (alguns com uso restrito). Recentemente (desde 2008), a entidade que organiza os nomes na ICANN, a GNSO (Generic Names Supporting Organization ou Organização para Suporte a Nomes Genéricos), montou a possibilidade de novos termos que poderiam ser utilizados por grandes organizações para identificar suas redes. Esses nomes podem ser cadastrados por organizações privadas. Exemplo de alguns do Brasil: .globo (Organizações Globo), .bradesco (Banco Bradesco), .ipiranga (Postos Ipiranga).
<b>ccTLD (Country Code Top Level Domain)</b>	Significa que o TLD está associado à abreviação de duas letras dos países.

Sigla	Descrição
<b>FQDN (Fully Qualified Domain Name)</b>	Sentença completa do domínio, envolvendo todas as partes: desde o subdomínio até o domínio de alto nível no fim do endereço.
<b>IDN (Internationalized Domain Names)</b>	No início, os termos usados nos endereços eram construídos usando um conjunto limitado de caracteres. Somente letras (a – z) e números (0 – 9), além do hífen, (-) podiam ser usados. Essa proposta permite a criação de domínios usando o idioma nativo do país. No Brasil, isso significa que podem ser criados domínios com letras acentuadas (ç, â).

Fonte: Elaborado pelo autor; ICANN (2016).

Com relação aos países, cada região tem sua organização, podendo incluir outros níveis de domínios até chegar à parte na qual a empresa ou organização pode registrar seu termo. Ao chegar ao ponto em que pode registrar seu próprio domínio, a empresa pode escolher um termo com até 63 caracteres.

Ao registrar um domínio para uso, deve-se identificar em qual segmento ele se enquadra (usando o gTLD ou ccTLD) e deve ser feita uma verificação se naquele nível não existe o mesmo nome já registrado. No Brasil, a entidade responsável por administrar os registros e padronizar os domínios de segundo nível é o Registro.br. Essa organização é parte do NIC.br, que é o órgão regulador da internet no Brasil. Segundo o próprio NIC (2016), “O Núcleo de Informação e Coordenação do Ponto BR – NIC.br foi criado para implementar as decisões e os projetos do Comitê Gestor da Internet no Brasil – CGI.br, que é o responsável por coordenar e integrar as iniciativas e serviços da Internet no País”.

Ao acessar o Registro.br, é disponibilizada uma interface para verificar se o domínio está disponível consultando a base de dados da região do Brasil. Na Figura 3.4, temos um exemplo de consulta de domínio (yrc.com.br) e o retorno da situação (o domínio não está disponível).

Figura 3.4 – Site do Registro.br com resultado de consulta de domínio



Fonte: Registro.br (2016).

No Brasil, após a indicação do país (.br), foram padronizadas várias categorias de domínios, além dos genéricos comuns a outros países (.com, .net, .org, .edu). Foram criados grupos para pessoas físicas e pessoas jurídicas. Há grupos restritos que precisam de comprovação para serem usados, como .mil, .br (usado pelas forças armadas) e .gov.br (para instituições governamentais).

Existem várias opções para profissionais liberais que indicam na sigla da abreviação do domínio uma semelhança com o nome da categoria profissional, como .arq.br (arquitetos) e .taxi.br (taxistas). O uso de domínios fáceis de serem lembrados e que geram rápida associação das pessoas com as empresas que os detêm levou à criação do projeto de novos domínios genéricos de primeiro nível. Esse projeto da GNSO foi criado para avaliar o impacto de novos nomes genéricos que podem ser cadastrados por empresas e ficariam diretamente associados a suas redes e seus servidores. Mais de mil empresas e associações já cadastraram seus domínios genéricos, inclusive algumas brasileiras, como .itau, .bradesco, .rio. Esses identificadores ficam no fim de um endereço de domínio.

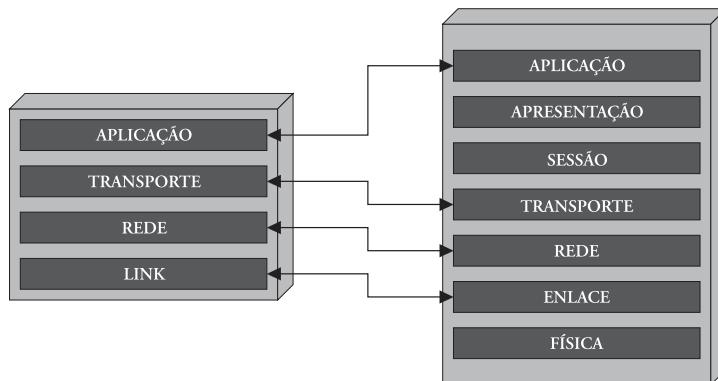
Com todo esse processo, empresas e pessoas físicas puderam ser encontradas mais facilmente dentro das redes, o que impulsionou o uso das redes e conexões entre-redes com a arquitetura TCP/IP como base.

### 3.3 Arquitetura TCP/IP

Como vimos, esse termo é uma junção de dois protocolos: TCP e IP. Primeiro foi criado o TCP, para estabelecer a comunicação com os poucos servidores no início da Darpanet. Depois foi criado o IMP (roteador) para verificar o pacote antes de chegar ao servidor e decidir se precisava ser encaminhado para outro roteador ou para o servidor da rede atual. Para isso foi criado um protocolo chamado IP, para tratar da primeira verificação sobre a qual rede o pacote de dados pertence.

Com a expansão dos serviços, ficou clara a necessidade de incluir novos protocolos para processar partes específicas da rede. Mas a estrutura utilizada para compor esse conjunto de funções foi simplificada em comparação com o modelo OSI. Com apenas quatro camadas, a estrutura do TCP/IP permite a comunicação entre os equipamentos e entre as redes.

Figura 3.5 – Comparação de camadas entre OSI e TCP/IP



Fonte: Elaborada pelo autor.

Apesar de não ter todas as camadas do modelo de referência, o TCP/IP reuniu todas as necessidades da transmissão com seus protocolos executando

as funções necessárias desde o empacotamento dos dados até a validação dos bytes do outro lado. Como o OSI é um modelo teórico, cada implementação real de protocolos pode definir quantas camadas quer utilizar para sua estrutura e desenvolvê-las de acordo com a necessidade.

Os protocolos são a chave da comunicação dentro do modelo TCP/IP. Com isso, verificaremos quais são os principais associados a suas camadas e assim direcionaremos melhor o entendimento.

### 3.3.1 Link (Interface de rede)

Na camada de link do TCP/IP, temos os protocolos utilizados para a comunicação física da rede. Nesse caso, essa camada apenas cria uma interface com os protocolos que estão relacionados ao hardware de transmissão. O Ethernet é um desses protocolos, como vimos no capítulo anterior.

Cada pacote Ethernet (802.3 ou 802.11) é recebido e processado. As informações são estruturadas para que a próxima camada (rede) identifique a parte do conteúdo que precisa usar para executar suas funções. Vários outros protocolos de acesso físico ao meio de transporte podem ser utilizados, mas vamos nos concentrar nos mais utilizados nas redes.

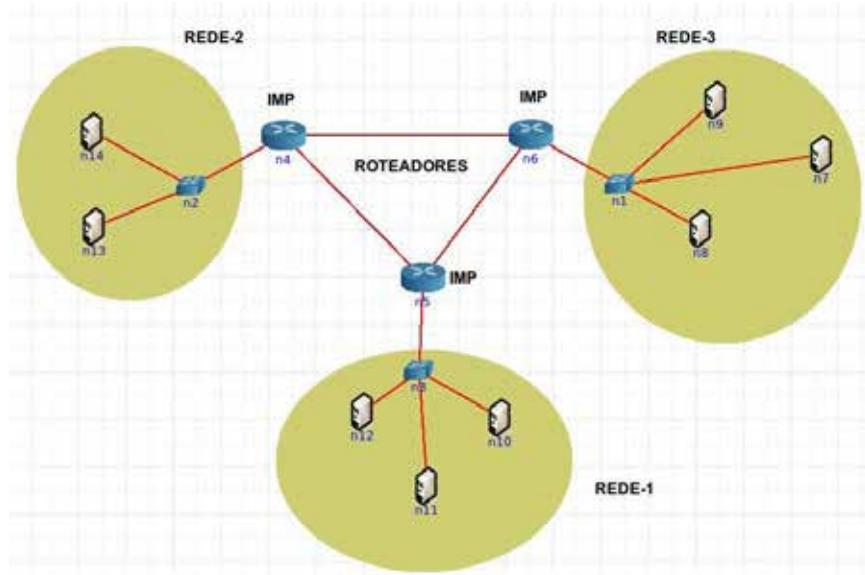
### 3.3.2 Rede

No período de construção da rede Arpanet, essa camada foi acrescentada para que os servidores pudessem avaliar os pacotes e encaminhar para o próximo ponto aqueles que não eram destinados a eles. Cada ponto teria as mesmas responsabilidades em encaminhar ou não os pacotes recebidos, não tendo a rede um ponto central de processamento.

Mas o recurso de processar para encaminhar os pacotes era dispendioso. Assim foi acrescentado um ponto de acesso antes do servidor para identificar se o pacote realmente era destinado ao servidor.

## Rede de Computadores

Figura 3.6 – Redes com roteadores (IMP) na entrada



Fonte: Elaborada pelo autor.

Criou-se, dessa maneira, um equipamento que acessava os pacotes até a camada de rede, identificava se o endereço pertencia à rede seguinte ou à própria rede. Somente depois dessa análise, deixava o pacote passar para o servidor. Esses equipamentos, chamados de IMP (*Interface Message Processor* ou Interface de Processamento de Mensagens), foram os precursores dos roteadores atuais.

A camada de rede do TCP/IP trata exatamente disto: endereços de destino são avaliados e então é identificada para qual rede o pacote deve ser direcionado. Esses endereços são a base do protocolo IP, que é o responsável pelas funções de roteamento. Assim, foram criados endereços de 32 bits que estariam associados a cada ponto da rede e identificariam o equipamento dentro da rede. Cabe ressaltar que os endereços IP são virtuais (endereços lógicos) e associados via software, não identificados na placa de rede.

Para uma grande rede como a da internet, não basta só encaminhar o pacote que pertence a outra rede. Existem algoritmos que foram criados

para avaliar a situação dos caminhos a serem utilizados. Esses códigos avaliam como está a velocidade do segmento de rede seguinte ou a quantidade de pontos intermediários até o receptor dos pacotes.

Ao receber o pacote, a camada de rede o envia para o processamento da camada de transporte.

### 3.3.3 Transporte

No TCP/IP, a camada de transporte é o berço do TCP. Esse protocolo foi desenvolvido para criar a comunicação entre duas máquinas no início da Darpanet, responsável por estabelecer a sessão e criar os controles de sequência e retransmissão dentro da arquitetura TCP/IP.

O TCP é o grande gerente da comunicação, que organiza os dados em partes para serem transmitidos e no outro lado remonta a informação. Caso perceba que um pacote tem algum problema (erro), estabelece a maneira de responder para o sistema de origem e “pede” que o pacote seja reenviado.

Outro protocolo que também se situa nessa camada é o UDP, mas nesse caso não há os mesmos cuidados do TCP com relação ao fluxo e à retransmissão, utilizado quando o desempenho é o maior objetivo. Assim que o recebimento dos pacotes estiver finalizado e a informação for remontada até sua forma original, os dados são repassados para a camada de aplicação.

### 3.3.4 Aplicação

Na última camada do conjunto TCP/IP, temos os protocolos que estão associados ao software propriamente dito que o usuário usa para se comunicar. Essa camada envolve diversos protocolos, cada um associado ao processamento de algum serviço, por exemplo, troca de arquivos, transmissão de vídeo ou mensagens. À medida que as pessoas sentem a necessidade de trocar informações de uma nova maneira, criam-se programas e protocolos para auxiliar na transmissão.

Após a camada de transporte identificar que o pacote está pronto, livre de erros e foi montado na sequência correta, o protocolo específico daquela comunicação entra em cena e retira os dados. Conforme o formato dos dados e

o tipo de comunicação feita pelo usuário, o protocolo pode simplesmente aceitar o pacote, pode gerar uma resposta para indicar o recebimento ou solicitar mais informações para dar continuidade ao serviço. O formato das informações pode envolver trechos de voz, arquivos de planilhas ou imagens de um site.

### 3.4 IPv6

Com as novas demandas de endereços para as redes e as empresas e com a participação destas na internet, houve um rápido esgotamento dos endereços inicialmente previstos na arquitetura TCP/IP. Os 32 bits reservados para gerar um número único para identificar os servidores (e demais dispositivos na rede) foram sendo distribuídos e, mesmo com algumas “manobras” utilizadas para estender o prazo, o endereçamento usado no protocolo IP extinguiu-se.

Essa situação já estava sendo avaliada pela comunidade da internet desde 1992, a partir de discussões iniciadas pela IETF (The Internet Engineering Task Force).

Como principal mudança, o IPv6 aumenta a capacidade de endereçamento de 32 bits para 128 bits. Com isso, todos os demais protocolos que usam o endereço antigo foram alterados para comportar o novo endereço.

#### Saiba mais

As documentações de protocolos, procedimentos e conceitos dentro da internet estão redigidas em RFC (*Request For Comments* ou Requisições para Comentários). Apesar de terem um cunho mais técnico, quando é necessário tirar dúvidas, podem ser acessadas gratuitamente na internet pelo site <<https://www.ietf.org/rfc.html>>.

### 3.5 Monitoramento de rede

Uma rede local ou a interligação de várias redes é objeto de muito cuidado pelos administradores de rede, envolvendo o acompanhamento do desempenho e o restabelecimento em caso de interrupção. Esses cuidados

estão cada vez mais em evidência, considerando todos os serviços que param em uma empresa quando a rede (ou o acesso à internet) é interrompido.

A fase de diagnóstico da interrupção de uma rede exige um esforço considerável do profissional de redes e pode ser feita com a investigação de quais partes da rede ainda estão comunicando ou com testes e respostas dos utilitários que estamos identificando em cada capítulo. Mas é importante saber quais são os parâmetros da rede em tempos normais para sabermos quando está acontecendo uma anomalia e assim conseguirmos tomar uma ação antes da interrupção total da rede.

Por exemplo, a rede está ocupando 70% da capacidade do switch central. Esse número é normal ou algum servidor está sendo sobrecarregado e a tendência é o colapso da rede? Sempre foi assim ou apenas pela manhã acontece a sobrecarga? Se olharmos os números isoladamente e de maneira esporádica, não temos como usar esses dados como referência caso aconteça algum tráfego anormal. Mas se pudermos ter os dados dos últimos dias ou meses, podemos perceber uma tendência que poderá ser administrada antecipadamente.

Outro ponto de análise ao monitorar uma rede é a quantidade de dispositivos que estão conectados e precisam ser avaliados e analisados periodicamente. Enquanto em redes pequenas basta acompanharmos um switch ou servidor, em redes maiores precisamos acompanhar uma quantidade maior e mais distribuída geograficamente; em muitos casos em outros prédios ou cidades.

E quando se pensa na frequência em que devem ser feitas as verificações, precisa-se analisar qual é a necessidade real da empresa, qual é a capacidade de avaliação e qual é o investimento que pode ser feito para isso. Avaliar um switch em um momento da manhã e outro à tarde não permite uma visão do tráfego que foi trabalhado ao longo do dia. Apenas uma fotografia de dois momentos. Caso essa verificação seja registrada por vários dias, nos mesmos horários, podemos ter uma visão histórica de como a rede tem se comportado no período nos dois momentos pontuais (um de manhã e outro à tarde) e permite trazer algum dado para uma análise parcial.

Para um monitoramento efetivo desse switch, seria importante coletar com um espaço de tempo mais curto entre cada captura dos dados (por exemplo, de 15 em 15 minutos) e assim ter uma percepção mais próxima da realidade do fluxo de dados processados durante o dia no equipamento. Bas-

taria criar um gráfico com os dados dentro de uma planilha e teríamos uma visualização dos momentos de maior ou menor utilização da rede. Se multiplicássemos a coleta de dados para mais equipamentos (mesmo que fossem selecionados só os mais críticos) já seria inviável fazer a coleta e o processamento desses dados manualmente.

Por isso, entram em campo os sistemas de monitoramento automatizados de rede, que trabalham automatizando a coleta de informações nos dispositivos e armazenando os dados em bases para poder criar pontos de alerta ou gráficos de volumes de bytes que entram e saem das interfaces. Veremos na sequência alguns sistemas de monitoramento OpenSource que possuem funcionalidades robustas e podem ser implementados sem custo em muitas empresas de pequeno e médio portes. Apesar de existirem sistemas de vários tipos, o principal objetivo é compartilhado pelos sistemas: centralizar em um ponto de visão único os principais equipamentos da rede e sua situação.

### 3.5.1 SNMP (Simple Network Management Protocol)

Protocolo utilizado para extrair informações remotamente de um aparelho conectado na rede. Com esse protocolo implementado no dispositivo na forma de um agente, os dados são coletados do hardware e armazenados em variáveis. A grande vantagem é a possibilidade de acessar essas variáveis com as informações da rede remotamente, principalmente considerando a dificuldade de acesso aos equipamentos em grandes redes distribuídas geograficamente.

Um sistema servidor consulta os agentes dos dispositivos e percorre a lista de variáveis disponíveis para selecionar a informação de que precisa: bytes das interfaces de rede, informações de memória, processador e algumas informações específicas do equipamento sendo monitorado. Essas informações são dependentes do equipamento e variam conforme a implementação do agente no dispositivo, podendo, por exemplo, ser relacionadas ao uso de disco em servidores, tensão de entrada em nobreaks ou ventilação em roteadores.

As informações de cada parte do hardware ficam estruturadas em uma base hierárquica chamada MIB (*Management Information Base* ou Base de Informações Gerenciais). Cada item coletado é um objeto dessa base.

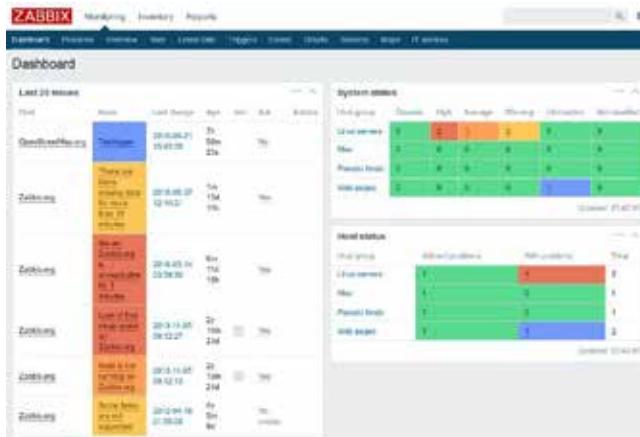
### 3.5.2 Zabbix

Ferramenta de monitoramento que acessa periodicamente todos os dispositivos configurados para avaliar se estão acessíveis e guardar a informação de cada acesso para histórico. O Zabbix coleta informações via SNMP e pode também utilizar outros utilitários para identificar se, além do acesso ao hardware e ao sistema operacional, é possível verificar se um serviço de rede, hospedado no dispositivo monitorado, está respondendo. Serviços como sites, e-mail, banco de dados e outros devem ser monitorados e enviar um alerta para o administrador caso não seja possível contato em uma ou mais tentativas.

Sua arquitetura é formada por um servidor que pode ser instalado em ambiente Linux (vários outros Unix também podem ser usados como ambiente base), em software agente instalado no computador a ser monitorado (pode também ser feita coleta remota dos dados sem agentes) e em base de dados. O acesso a switches e roteadores pode ser feito através do protocolo SNMP, assim como muitos sistemas operacionais de servidores que também possuem serviços de comunicação por meio desse protocolo. As informações são verificadas em um browser que pode estar fora da rede monitorada.

Na figura 3.5 temos um exemplo do funcionamento do Zabbix usando o monitoramento de demonstração feito no próprio site no qual o sistema é hospedado.

Figura 3.7 – Tela de demonstração de monitoramento do Zabbix



Conforme indicado em Zabbix (2016), o sistema de monitoramento tem algumas funções bem abrangentes para o monitoramento da rede e de seus componentes. Pode ser utilizada para monitorar servidores (se estiverem em funcionamento), mas também pode ser usado para identificar se um serviço específico está em funcionamento. A diferença é que um equipamento pode estar ligado e seu sistema operacional pode estar conectado e respondendo a uma verificação, mas isso não quer dizer que o serviço que estaria compartilhado um recurso (por exemplo, compartilhamento de arquivo, banco de dados, sites) esteja respondendo a consultas específicas a suas funções. Nesse sentido, o Zabbix pode construir e monitorar o acesso aos recursos de forma personalizada, ou seja, podem ser construídas formas de acesso que são incluídas na base de monitoramento com a construção de gráficos e séries históricas.

Como forma adicional de visualização da rede, o Zabbix pode construir um mapa com os dispositivos encontrados e a comunicação entre eles. Também possui um sistema interno de abertura de chamados de suporte da TI.

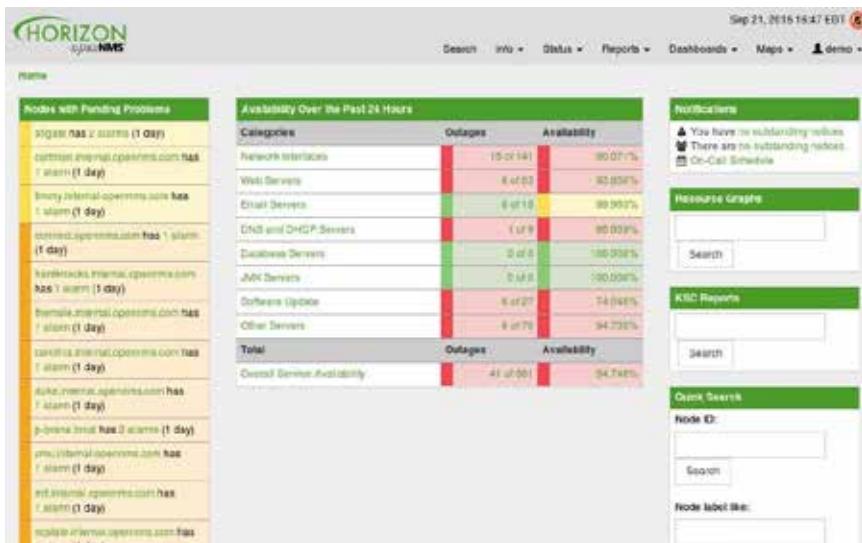
### 3.5.3 OPENNMS

Esse sistema, semelhante ao Zabbix, permite a construção de um conjunto de objetos a serem monitorados por acesso remoto ao dispositivo ligado à rede.

Segundo OPENNMS (2016), esse framework possui formas de integração com sistemas de chamados, plataformas de envio de notificações (e-mail, SMS) e alarmes. Mantém as informações em um banco de dados PostgreSQL e, para grandes coletas de dados, utiliza a base Cassandra.

Na figura 3.8, temos a tela inicial do sistema em funcionamento no próprio site do projeto, com o monitoramento de diferentes pontos e serviços.

Figura 3.8 – Tela inicial do sistema com monitoramento dos servidores do projeto OPENNMS



Podemos verificar os pontos da rede com problemas, as estatísticas de disponibilidade separadas por categorias (servidores web, servidores de e-mail) e o histórico de cada dispositivo para avaliar as interrupções.

O acesso às informações é feito via browser e a coleta delas pode ser feita através de agentes (pequenos programas que enviam as informações para o servidor) ou remotamente com o acesso ao serviço que o equipamento disponibiliza. O uso do protocolo SNMP também faz parte do conjunto de possibilidades.

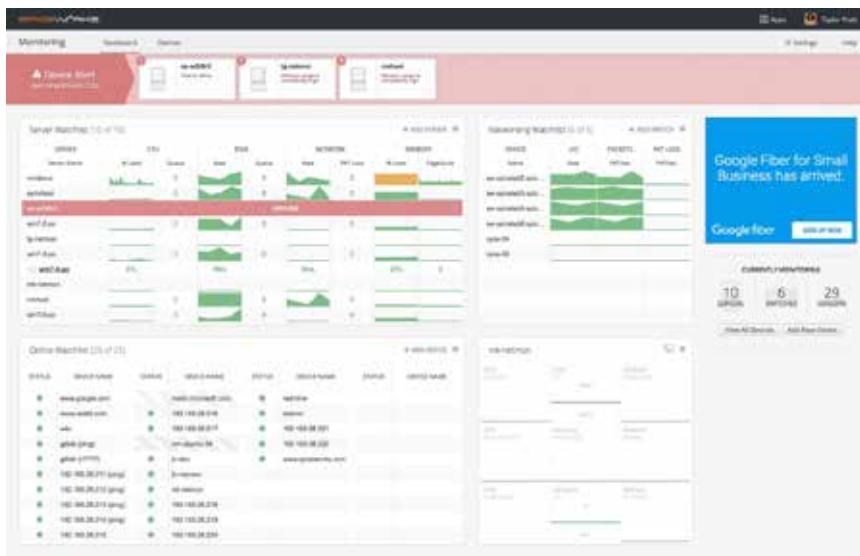
Como o sistema Opennms é desenvolvido em Java, pode ser instalado tanto em Linux quanto em sistemas Windows.

### 3.5.4 Spiceworks

O sistema Spiceworks é uma plataforma dirigida ao sistema Windows, mas também permite monitorar sistemas Linux. Conforme Spiceworks (2016), o sistema pode ser configurado para enviar alarmes em caso de interrupção de serviços, fazer a coleta de dados de switches e roteadores via SNMP, acessar informações de desempenho dos computadores e manter um inventário das máquinas conectadas na rede. Além da parte da rede em que os equipamentos estão conectados, o Spiceworks pode acessar máquinas Windows ou Linux com o usuário administrador e fazer uma coleta de softwares instalados para a geração de relatórios.

Na Figura 3.9, temos uma tela usada no site do projeto para demonstrar as informações que podem ser vistas na interface do sistema. Essas informações são visualizadas por meio de um browser e também podem ser vistas em um aplicativo instalado no smartphone do administrador de rede.

Figura 3.9 – Tela de demonstração do monitoramento feito pelo Spiceworks



O sistema está relacionado a uma comunidade de profissionais de TI envolvidos na área de suporte que podem trocar experiências sobre o uso do sistema e características técnicas dos dispositivos monitorados.

### 3.5.5 NTop

Diferente dos sistemas vistos, o NTop é um programa instalado diretamente no servidor a ser monitorado. Com esse serviço, são analisadas todas as conexões que entram e saem do servidor à medida que vão acontecendo, criando um quadro geral da situação com a distribuição dos bytes entre os protocolos.

Pode ser instalado em servidores Windows e Linux e gera gráficos com dados históricos do tráfego dos servidores e dos equipamentos que estão associados a eles. Caso seja preciso verificar dados detalhados do equipamento que está se comunicando com o servidor sendo monitorado, é possível clicar no endereço do equipamento e ver pontos intermediários entre os dois, quantidade de dados trocados e com qual serviço ou em qual período do dia houve a maior parte do tráfego.

Como podemos perceber, alguns dados são iguais aos já vistos com os utilitários Netstat ou Traceroute, mas são feitas agregações e segmentação das informações em uma interface web fácil de navegar. E os dados são armazenados para serem acessados posteriormente.

O NTop pode ser deixado como um serviço no servidor que estará colecionando, analisando e gravando as informações para quando o administrador do servidor precisar avaliar a situação.

## Rede de Computadores

Figura 3.10 – Gráfico de acessos da última hora no NTop mostrando os principais acessos do servidor



Fonte: Elaborada pelo autor.

Em alguns casos, o NTop pode identificar a localização geográfica do equipamento que está conectado ao servidor e avaliar se o serviço web que está sendo acessado tem algum tipo de malware de “lista negra” da internet, como o Google Safe Browsing (<https://developers.google.com/safe-browsing>) ou o Http:BL do projeto Honey Pot (<http://projecthoneypot.org/>).

## Síntese

Tivemos neste capítulo uma visão do início da internet, que é uma grande rede que interliga outras redes. Pudemos analisar como os protocolos são estruturados com a característica de cada camada e como é facilitado o acesso a dispositivos utilizando nomes ao invés de endereços numéricos difíceis de guardar na memória. Para isso, temos diversas organizações internacionais com a responsabilidade de administrar as decisões envolvendo países

de diferentes idiomas e com diferentes infraestruturas de rede. No Brasil, o Registro.br disponibiliza ferramentas que permitem identificar se é possível utilizar um nome para uma empresa ou pessoa física; se o nome estiver disponível, há uma estrutura pronta para administrar essas questões.

Tivemos a introdução à arquitetura TCP/IP e vimos que na verdade há vários protocolos trabalhando em conjunto, cada um com suas funcionalidades. Percebemos que a situação dos endereços usados na internet já se esgotou e existe uma nova versão caminhando para ser disseminada.

Por último, tivemos a oportunidade de avaliar algumas opções de sistemas que podem auxiliar no monitoramento das redes.

## Atividades

1. No conjunto de protocolos TCP/IP, temos algumas camadas com funções e protocolos específicos. Uma delas está relacionada a um protocolo já visto em capítulos anteriores e ao meio de transporte utilizado na transmissão entre um ponto e outro. Indique qual é o protocolo e cite em qual camada do TCP/IP ele está relacionado e por quê.
2. Responda:
  - a) No início da Darpanet, para evitar que o servidor precisasse avaliar cada pacote que recebesse e dispendesse parte de seu processamento encaminhando esse pacote caso fosse de outro servidor, foi desenvolvido um equipamento cuja funcionalidade é utilizada até hoje. Cite o nome do equipamento na época e o nome do equipamento atual que realiza a mesma função hoje em dia.
  - b) Qual foi o problema no protocolo TCP/IP que obrigou o desenvolvimento da nova versão do protocolo IP (IPv6)?
  - c) Qual foi o motivo que levou a utilizar as letras dos países na construção dos nomes de domínio na Internet?
3. Um administrador de rede cuida de vários dispositivos em uma empresa que tem escritórios em dois andares de um prédio. Muitas

vezes, precisa saber se o switch do outro andar está com a capacidade esgotada. Qual protocolo pode ser usado para auxiliar nessa tarefa e por quê?

4. Um dos protocolos apresentados no modelo TCP/IP é responsável por organizar o fluxo de pacotes sendo transmitidos, avaliar se existe erro, solicitar a retransmissão e separar os dados em partes para serem enviados pelas camadas inferiores. Cite o nome do protocolo e qual é a camada a que pertence.

# 4

## Rede

NESSE CAPÍTULO VAMOS estudar os conceitos e estrutura da camada de rede da arquitetura TCP/IP. Nesta camada, como vimos em capítulos anteriores, temos um dos principais protocolos: o IP (*Internet Protocol*). Entraremos em detalhes sobre como esse protocolo trabalha e veremos na prática seu funcionamento por meio de algumas ferramentas. Analisando a estrutura do protocolo IP, podemos entender outras características do seu novo formato, o IPv6.

A ESTRUTURA DO protocolo IP em conjunto com equipamentos e protocolos de roteamento, encontram o dispositivo de destino. Esse dispositivo pode estar associado a um nome, para facilitar o acesso, e pode estar em qualquer outra rede. As redes podem estar também associadas a um nome de domínio. Nesse capítulo, estaremos mecanismos para trocar os nomes de máquina e de rede para

um endereço numérico, além de um mecanismo para associar o endereço de hardware a um endereço virtual. Assim, conseguimos passar por vários segmentos de redes até chegar ao destino final.

## 4.1 IPV4

No início da Darpanet, precursora da Internet, eram poucos e caros os servidores que estavam conectados em rede e seu processamento era dividido com vários usuários sendo atendidos via terminais. Ou seja, o processamento de todas as tarefas era feito no próprio servidor.

### Saiba mais

Nesse período, os servidores tinham características bem diferentes dos atuais. Atualmente voltou-se a utilizar o conceito de terminais, mas naquela época este era o único formato de acesso ao servidor (*mainframe*). Pesquise sobre sistemas operacionais e compartilhamento de processamento, para conhecer o contexto dos grandes servidores da época.

Com o crescimento da rede Darpanet, houve a necessidade de melhorar o desempenho da rede e separar o processamento dos encaminhamentos dos pacotes do processamento das tarefas normais do servidor. Ao ficar diretamente conectado na rede, uma parte do processamento do servidor fica relacionada à verificação dos endereços de destino dos pacotes, e encaminhamento para o próximo servidor quando este endereço não coincide com o seu.

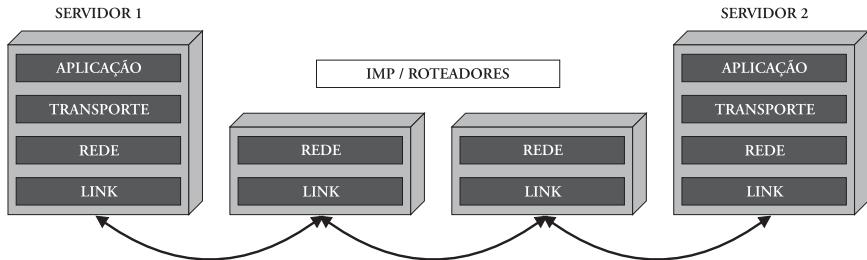
A construção do IMP (precursor dos atuais roteadores), possibilitou que este processamento fosse feito fora do servidor, num equipamento dedicado, e que trabalhe com as duas camadas iniciais: *enlace* e *rede*.

Desta forma, os roteadores fazem o trabalho de verificar quais pacotes pertencem à rede local e quais devem ser enviados para outra rede ou para o próximo roteador, que também fará o mesmo tipo de verificação, até que

o pacote encontre o roteador que está conectado à rede em que o servidor solicitado no endereço destino esteja associado.

Na figura 4.1, temos o trajeto percorrido pelo pacote através dos roteadores até chegar no seu destino.

Figura 4.1 – Trajeto do pacote através dos roteadores



Fonte: Elaborada pelo autor.

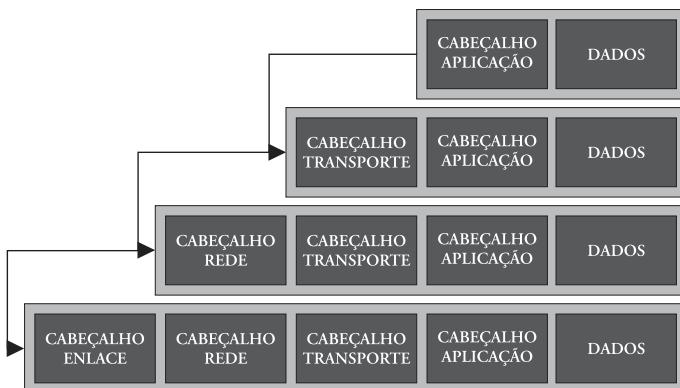
Pode haver também diferentes meios físicos entre os roteadores e os servidores. Podemos ter Ethernet na conexão entre o roteador e o servidor, e também podemos ter uma linha telefônica ou uma transmissão via satélite. Podemos inclusive ter vários meios físicos conectados no mesmo roteador ao mesmo tempo, uma vez que o roteador pode ter duas ou mais interfaces de rede e, portanto, conectar duas ou mais redes diferentes.

É importante notar também que, apesar de no início da Darpanet ter sido desenvolvido um equipamento específico como IMP e nos dias atuais termos os roteadores na borda de nossas redes, qualquer máquina que tenha mais de uma interface pode também atuar como um roteador. Basta ter o software apropriado para encaminhar os pacotes entre as suas interfaces de acordo com os requisitos do protocolo.

Para que os pacotes possam ser analisados pelos roteadores e demais dispositivos de maneira rápida, as informações de parâmetros dos protocolos ficam agrupadas no início de cada protocolo sendo utilizado. A este grupo de parâmetros chamamos *cabeçalho*.

Desta forma, cada protocolo que é acrescentado na conexão acrescenta seus parâmetros antes dos dados ou protocolos anteriores, num processo chamado de *encapsulamento*.

Figura 4.2 – Processo de encapsulamento de protocolos



Fonte: Elaborada pelo autor.

#### 4.1.1 Cabeçalho do protocolo IP

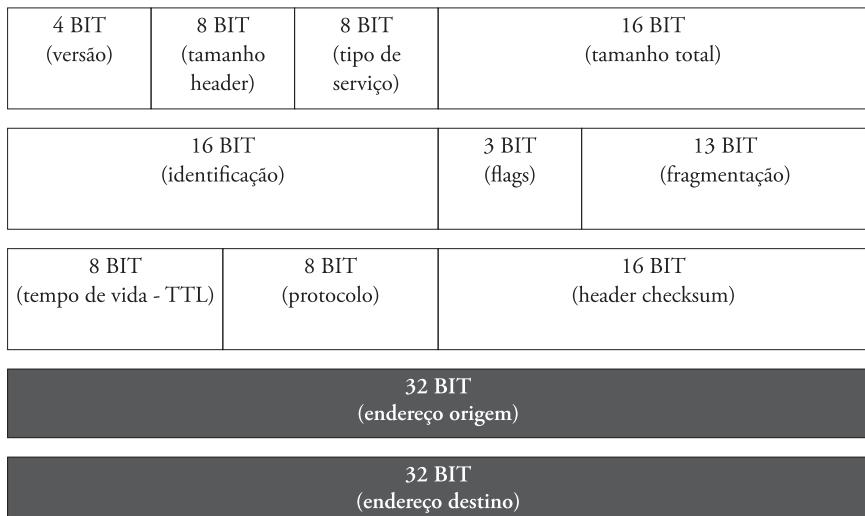
A cada informação a ser enviada é necessário configurar alguns parâmetros aos equipamentos, para que possam identificar qual caminho deve ser utilizado e como este trajeto deve ser feito. Estes parâmetros são anexados ao pacote de dados e possuem uma estrutura padronizada. Veja a figura 4.3 a seguir, na qual temos a quantidade de bits usados para o armazenamento das informações. Nesta figura, temos a primeira informação, que é a versão do protocolo IP que está configurado na sequência. Desta forma, mesmo que haja uma mudança estrutural no cabeçalho do protocolo, é possível que as versões possam coexistir na mesma rede. Os equipamentos, ao verificarem o cabeçalho e separarem a informação de qual é a versão, podem se adaptar para ler os bits na sequência que esta versão pede. Considerando que temos 4 bits para a identificação do protocolo, podemos ter um valor de 0 a 15. Na versão atual, temos o valor 0100 (quatro) e na nova versão do IP temos o valor 0110 (seis).

Depois da versão do protocolo, vem o tamanho do cabeçalho. Este campo de 4 bits indica quantos conjuntos de 32 bits podem ser utilizados no total, neste caso, 480 bits (ou 60 bytes).

Na figura 4.3 temos cinco conjuntos de bits (veja as informações na horizontal, agrupadas em grupos de 32 bits). Este é o cabeçalho mínimo

usado pelo protocolo IP, num total de 160 bits. Após os dois endereços de 32 bits, podem ser acrescentadas informações opcionais num máximo de 320 bits. Estes dois números somados chegam no máximo de valor que os 4 bits podem ser combinados.

Figura 4.3 – Cabeçalho IPv4



Fonte: Elaborada pelo autor.

O próximo campo, chamado de *tipo de serviço* ou também de *qualidade de serviço*, começou a ser utilizado em tempos mais recentes, para identificar qual a priorização a ser feita com o pacote.

No campo *tamanho total* do pacote, com 16 bits podemos chegar a um valor de até 65.535 bytes, indicando o máximo que um pacote IP pode ter. Apesar deste valor, deve-se perceber que os protocolos da camada de ENLACE podem ter um tamanho diferente. Isto é o que acontece por exemplo com o Ethernet, onde o tamanho máximo chega a 1.500 bytes, como vimos no capítulo 2.

Por este motivo, muitas vezes é preciso “quebrar” (segmentar) o pacote de dados em mais de um pedaço. E, mais importante ainda: precisamos montar tudo de novo no outro lado. Para auxiliar nessa tarefa, temos o campo de *identificação* de 16 bits. É calculado um valor que deve ser o mesmo para todos os pacotes que pertencem à mesma informação que foi dividida. Ao

chegar do outro lado, o roteador verifica se o pacote tem o mesmo código de identificação e, com isso, ele sabe que o pacote pertence à mesma sequência que está sendo recebida. Esse código, em conjunto com o campo de 16 bits de *fragmentação*, controla esta situação de divisão do pacote IP. O campo de fragmentação indica a posição que os bytes ocupam dentro do pacote, para auxiliar a remontagem depois de recebidos.

Os bits de *flags* são usados para o controle da fragmentação, indicando se o pacote pode ser fragmentado e quando for o último fragmento.

O campo TTL (*time to live*) indica o máximo de tempo que o pacote pode ficar “viajando” dentro da rede. Este número é decrementado por todos os roteadores no meio do caminho e, ao chegar em zero, o roteador descarta o pacote e manda uma mensagem ao roteador de origem.

O campo *protocolo* indica qual código do protocolo da camada de transporte está sendo encapsulado no datagrama, e o campo *header checksum* é um cálculo feito com os campos do cabeçalho para verificar se houve alguma alteração nas informações. Caso haja algum erro na transmissão, o número calculado no roteador difere do número deste campo e permite que o erro seja detectado.

Por fim, temos os endereços de 32 bits da origem e destino.

#### 4.1.2 Endereço IP

Os dois últimos campos do cabeçalho padrão do protocolo IP são os que permitem a transferência da informação pelas redes, chegando a até 4.294.967.296 combinações de endereços possíveis ( $2^{32}$  bits).

Para informar o endereço de origem e destino, foi criada uma notação que permite a manipulação dos endereços mais facilmente por pessoas. Esta notação segue o seguinte formato: **X.X.X.X**, onde cada X é um conjunto de 8 bits separados por um ponto. São quatro conjuntos de 8 bits, somando no total os 32 bits dos campos de endereço de origem e de destino. Cada conjunto de 8 bits pode ter um valor de 0 a 255 (decimal). Portanto, os endereços podem iniciar em 0.0.0.0 até o endereço máximo de 255.255.255.255 e não podem ser duplicados, ou seja, cada endereço distribuído deve ser utilizado **apenas uma vez** em uma interface de rede (a mesma lógica utilizada nos

endereços *Mac Address* da Ethernet). Nada impede alguma interface de ter mais de um endereço IP ou de haver uma troca de endereços entre interfaces, mas o mesmo endereço público não pode estar em mais de uma interface.

O endereço IP é hierárquico e, por isso, está organizado em **duas partes**: uma parte indica a sequência da rede em que o endereço está associado, e a segunda parte é a sequência que fica associada à máquina dentro da rede.

Mas nem todos os endereços possíveis foram distribuídos. Algumas sequências são separadas para uso restrito ou para indicar uma situação pré-configurada. Um exemplo é a faixa 127.0.0.0, que é utilizada para a comunicação interna do sistema operacional. Com esta faixa, o servidor pode enviar e receber informação no mesmo padrão de comunicação da Internet, sem que os dados sejam enviados ao meio de transporte físico da rede. Um endereço desta faixa é utilizado também para testes da interface de rede. Ao acessar o endereço 127.0.0.1, estamos nos comunicando com a própria máquina, fazendo com que os bytes enviados não saiam do servidor. No quadro 4.1 temos um exemplo do comando ping, já apresentado no segundo capítulo, com o uso do endereço de teste mostrado.

Quadro 4.1 – Exemplo de comando ping (GNU/Linux)

```
1. ping -c 4 127.0.0.1
2.
3. PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
4. 64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.042 ms
5. 64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.078 ms
6. 64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.069 ms
7. 64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.067 ms
8.
9. --- 127.0.0.1 ping statistics ---
10. 4 packets transmitted, 4 received, 0% packet loss, time 2999ms
11. rtt min/avg/max/mdev = 0.042/0.064/0.078/0.013 ms
```

Fonte: Elaborado pelo autor.

No quadro 4.1, temos a sintaxe do comando (linha 1), o resultado de cada resposta (linhas 3 a 7) e uma estatística sobre todo o conjunto. Nesta estatística, temos a quantidade de pacotes enviados e recebidos (linha 10) e uma avaliação dos tempos de resposta dos pacotes. Nesta avaliação de tempos, temos o menor tempo dos pacotes enviados (min), seguido pela média dos 4 pacotes enviados (avg em negrito) e o tempo da maior demora de

resposta (max). Por fim, temos o cálculo do desvio dos tempos de resposta (mdev) em relação à média.

No Linux (usado no exemplo acima), temos que indicar quantos pacotes serão transmitidos usando a opção `-c 4` e no Windows ele apenas envia os quatro pacotes e interrompe sozinho.

Outro exemplo de faixas de endereços reservadas são as faixas de IPs utilizadas para a configuração de redes internas. São os chamados endereços privados. No quadro 4.2 temos um resumo destes endereços.

Quadro 4.2 – Conjunto de faixas de endereços privados

End. Início	End. Fim	Qtd. Endereços
10.0.0.0	10.255.255.255	16.777.216
172.16.0.0	172.31.255.255	1.048.576
192.168.0.0	192.168.255.255	65.536

Fonte: Elaborado pelo autor.

Esses endereços privados foram separados do grande grupo devido à crise de endereços IPs à medida que eram esgotadas as reservas mundiais. Como as empresas tinham muitas máquinas desktop que não tinham a função de compartilhar ou disseminar informações para a grande rede, os endereços públicos utilizados nestas máquinas ficavam desperdiçados, ao mesmo tempo que a falta de endereços criava cada vez mais restrição na distribuição das faixas.

## 4.2 NAT (Network Address Translation)

Para resolver o dilema, foi desenvolvido um mecanismo chamado de NAT (*Network address translation* ou, traduzido, *endereços de rede*), que permitia o uso de **um** único endereço IP público mesmo que a empresa tivesse centenas de equipamentos na rede interna. Este mecanismo era executado num equipamento que fica na borda da rede da empresa e, ao perceber a saída

de um pacote para a Internet, troca o endereço de origem para um endereço público da empresa e armazena a informação em uma tabela. Ao retornar a resposta, este equipamento volta à informação do endereço original e repassa para a máquina interna que solicitou a conexão. Com este mecanismo funcionando na rede da empresa, os endereços internos que eram utilizados para a conexão entre as máquinas internas não precisavam ser visíveis na Internet e, portanto, não precisavam ser únicos ou roteáveis.

Isso permitiu que os endereços reservados mostrados no quadro 4.2 pudessem ser utilizados na rede interna de várias empresas, sem o risco de duplicidade na Internet e, consequentemente, diminuindo a demanda de endereços públicos para equipamentos que não compartilhavam informação. Caso o equipamento apenas precisasse acessar algo na Internet (que é a necessidade de grande parte dos desktops), ele simplesmente passava pelo servidor que processa o NAT.

### 4.3 Máscara de rede (Network mask)

Com a distribuição inicial dos endereços de rede, foram feitos grupos (chamados classes) com uma quantidade pré-definida de endereços e a indicação de quais valores pertenceriam a uma rede. Este formato não se sustentou, a partir do momento em que uma grande quantidade de endereços eram distribuídos e havia muito desperdício de endereços nos grupos e, em alguns casos havia a necessidade de subdividir os endereços internamente nas empresas.

A partir destas dificuldades, foi desenvolvido um método mais flexível para identificar a parte da rede no endereço IP. Em vez de utilizar um formato fixo, como eram as classes, é combinada junto com o endereço IP uma máscara de 32 bits que, por meio de um cálculo entre os dois, permite separar a parte da rede da sequência interna do endereço. Este número de máscara serve para indicar quantos bits, a partir da esquerda, serão usados para o endereço de rede. No quadro 4.3 temos as duas notações usadas para a máscara de redes.

Quadro 4.3 – Formato de máscaras

Notação	Descrição
255.255.255.0	Os bits que estarão ligados são informados através de 4 conjuntos de 8 bits em decimal. 255 representa em decimal que os 8 bits estão ligados (11111111) e precisa ser informado em ordem, da esquerda para a direita. Para indicar uma quantidade de bits ligados diferente, basta indicar o decimal correspondente no fim da sequência. Por exemplo: 255.255.224.0 ( 11111111.11111111.11100000.00000000 ) ou 255.255.240.0 ( 11111111.11111111.11110000.00000000 ).
/24	A colocação de uma barra e um número após o endereço IP indica quantos bits da máscara da esquerda para a direita estarão ligados. O máximo que podemos ter é 32, indicando que todos os bits estão ligados. Por exemplo: um endereço 192.168.1.0/16 indica que os primeiros 16 bits da máscara estão ligados e é o equivalente a 255.255.0.0 ou 11111111.11111111.00000000.00000000.

Fonte: Elaborado pelo autor.

As duas notações mostradas podem ser utilizadas, basta que o sistema que esteja sendo configurado indique qual delas consegue interpretar, ou se utiliza ambas.

O cálculo feito entre o endereço IP e a máscara envolve uma operação binária AND e o resultado é a parte da rede. No quadro 4.4 temos um exemplo de identificação da parte da rede.

Quadro 4.4 – Cálculo AND binário

	Decimal	BITS
IP	192.168.1.20	11000000.10101000.00000001.00010100
MASC	255.255.255.0	11111111.11111111.11111111.00000000
<b>RESULTADO</b>	<b>192.168.1.0</b>	<b>11000000.10101000.00000001.00000000</b>

Fonte: Elaborado pelo autor.

No quadro anterior, a primeira linha indica o endereço IP do qual queremos descobrir a parte da rede com as duas opções de visualização: decimal e em bits. Na segunda linha temos a máscara a ser usada para separar a parte da rede. Na terceira linha temos o resultado do AND binário entre o endereço IP e a máscara. Nesta operação, caso o bit do IP e o bit da máscara seja 1, então o resultado é 1. Caso tenhamos zero no IP ou na máscara, o resultado será zero. Compare os bits do endereço IP e os bits da máscara e veja o resultado correspondente na terceira linha.

Os bits que sobram da máscara (no exemplo acima 255.255.255.0 é o mesmo que /24), no caso os 8 bits restantes (32 bits – 24 bits da máscara = 8 bits), são os endereços que podem ser usados nas interfaces. Nesse exemplo, 8 bits é o mesmo que 256 endereços ( $2^8$ ). Na prática, reservam-se o primeiro e o último endereço para representar, respectivamente, a rede e a comunicação em *broadcast*, ficando assim com 254 endereços úteis.

Com tudo isso, podemos configurar uma interface de rede com o endereço IP único que deve existir na rede e a máscara, que permitirá verificar se o endereço de destino que precisamos acessar faz parte da mesma rede. Mas estas verificações estão relacionadas ao TCP/IP. Do ponto de vista físico, como uma máquina percebe os endereços IPs de outras máquinas?

Vimos em outros capítulos que numa rede Ethernet, as interfaces de rede já vêm com um endereço de 48 bits marcado na placa. E é este endereço que é pesquisado por meio de switchs e outras interfaces de rede.

Neste ponto, chegamos em uma encruzilhada: temos que encontrar um outro endereço IP na rede para conectarmos, mas a rede local é capaz de mapear apenas os endereços de hardware (*mac address*).

### 4.3.1 ARP – Address Resolution Protocol

Para podermos trabalhar com o TCP/IP numa rede real, temos que descobrir qual *mac address* está associado a cada endereço IP.

No capítulo 2, tivemos alguns exemplos de captura de tráfego do protocolo ARP, que faz exatamente a tarefa que vimos acima. Para isso, o sistema operacional, quando quer se comunicar com outro endereço IP, lança na rede Ethernet um pacote com o endereço *mac* de destino com todos os bits ligados (*broadcast*) e, com isso, todas as máquinas que estão conectadas à rede acabam recebendo essa solicitação. Este pacote especial tem internamente o endereço IP que a máquina de origem precisa se conectar.

Todas as máquinas inspecionam o pacote e verificam o endereço IP que está contido na parte de dados. Se a máquina não possui o IP, simplesmente descarta. Quando a máquina que possui aquele IP recebe também o pacote (via *broadcast*), ela responde para a máquina de origem com o seu *mac address* na resposta.

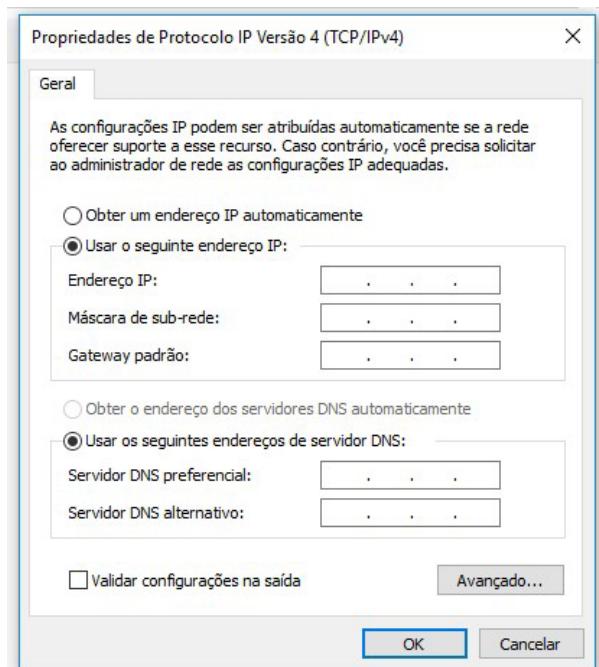
Desta forma, a máquina de origem consegue o *mac address* do equipamento que precisa fazer a conexão e cria um pacote Ethernet com o cabeçalho IP apropriado, enviando via interface de rede.

E para finalizar esta sequência lógica do endereçamento, fica faltando apenas a configuração do IP na máquina. Existem maneiras de parametrizar a interface (ou interfaces) do equipamento: manualmente, digitando o endereço IP e a máscara, ou dinamicamente, executando uma rotina que automataiza a seleção de um IP.

Na versão manual, apesar de a operação poder variar conforme o sistema operacional e o dispositivo (alguns dispositivos como por exemplo as impressoras, tem um pequeno display para esta digitação) na prática, ao chegar na tela para a colocação do endereço, são disponíveis espaços para a digitação: do IP, da máscara, do endereço do *default gateway* e o endereço do servidor de DNS (estes dois últimos veremos daqui a pouco).

Ao terminar a digitação, a interface fica configurada com os respectivos endereços e pode iniciar a comunicação. Na figura 4.4 temos um exemplo da tela de configuração no sistema Windows.

Figura 4.4 – Tela de configuração de IP no Windows 10



Ao finalizar a digitação dos endereços na interface de rede, podemos iniciar a conexão com outras máquinas com a indicação deste endereço IP como origem.

O *default gateway* ou *gateway padrão* é o equipamento que irá interligar a rede com outras redes e, na grande maioria das redes, este é o papel do roteador. Mas também temos o firewall (dispositivo de segurança e filtragem de pacotes) executando esta função. Normalmente ele é a porta de ligação com a Internet, mas em grandes redes é possível ter roteadores internos e mais de uma faixa de rede sendo utilizada internamente. Nestes casos, o *gateway* será o equipamento que irá transferir os pacotes entre as redes.

Para que isso funcione, o sistema operacional sempre verifica o endereço IP de destino e compara a parte da rede com a parte da rede do IP que está configurado em sua placa de rede. Caso sejam iguais, isto quer dizer que o endereço de destino também pertence à rede local. Caso sejam diferentes, quer dizer que o endereço de destino pertence a outra rede e precisa de um auxílio para sair da rede local e verificar onde está a rede e o equipamento que precisa se conectar.

Mas em redes maiores, podemos automatizar esta configuração de endereços e ainda auxiliar no gerenciamento dos endereços utilizados internamente. Caso seja configurado apenas na forma manual, é preciso organizar um método de liberação dos IPs, para que não haja engano e seja cadastrado um mesmo número de IP em mais de um dispositivo. Este controle acaba sendo complexo, quando é acompanhado do cotidiano movimentado de uma empresa, com *notebooks* entrando e saindo da rede, impressoras e servidores que precisam ser encontrados facilmente pelos usuários, troca de equipamento por problema de hardware e o uso de *tablets* e *smartphones* que podem acessar a internet pela rede corporativa.

#### 4.3.2 DHCP – Dynamic Host Configuration Protocol

Este protocolo vem de encontro a este cenário, em que a administração do uso e distribuição dos endereços de IP pode ser executada por um servidor da rede.

Para exemplificar como esta mágica acontece, vamos ver como uma máquina inicia sua vida na rede TCP/IP.

- a) Ao ser ligada, a máquina possui apenas o endereço *mac address Ethernet*. Por meio dele envia um pacote em *broadcast* para todas as máquinas solicitando um endereço IP para ser usada em sua interface.
- b) Um servidor DHCP recebe também o pacote em *broadcast* e verifica que é um pedido de endereço IP.
- c) Verifica então na sua lista se existe algum endereço IP liberado para uso.
- d) Caso tenha, marca ele como usado, cria um pacote Ethernet com o endereço IP dentro dele e envia para o *mac address* de

quem solicitou, fazendo assim um “emprestimo” deste IP por um tempo limitado.

- e) A máquina que solicitou recebe o pacote com seu *mac address* e utiliza o IP enviado para configurar sua placa de rede.

Figura 4.5 – Wireshark com a sequência de solicitação DHCP

No.	Time	Source	Destination	Protoc	Len/	Info
8	12.755759 0.0.0.6	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x486cb92d	
11	13.7577231 192.0.2.10	192.0.2.11	DHCP	342	DHCP Offer - Transaction ID 0x486cb92d	
12	13.7577330 0.0.0.6	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0x486cb92d	
13	13.764986 192.0.2.10	192.0.2.11	DHCP	342	DHCP ACK - Transaction ID 0x486cb92d	
Frame 8: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits)						
Ethernet II [Src: 00:00:00 aa:00:00 (00:00:00:aa:00:00)   Dst: Broadcast (ff:ff:ff:ff:ff:ff)]						
Destination: Broadcast (ff:ff:ff:ff:ff:ff)						
Source: 00:00:00 aa:00:00 (00:00:00:aa:00:00)						
Type: IP (0x0800)						
Internet Protocol Version 4, Src: 0.0.0.0 (0.0.0.6), Dst: 255.255.255.255 (255.255.255.255)						
User Datagram Protocol, Src Port: bootpc (68), Dst Port: bootps (67)						
Bootstrap Protocol						

Na figura 4.5, temos a mensagem capturada via *Wireshark* dentro de uma rede simulada com *CORE*. Estas duas ferramentas foram vistas no capítulo 2. Na tela capturada, podemos ver em destaque a linha do protocolo Ethernet, no qual temos um *mac address* de origem (**00:00:00:aa:00:00**) e o *mac address* de broadcast (**ff:ff:ff:ff:ff:ff**).

A resposta do servidor DHCP pode ser vista no quadro a seguir, no qual foi usada também uma ferramenta vista no capítulo 2 para capturar as mensagens, dentro da mesma rede simulada com *CORE*.

Quadro 4.5 – Trecho de captura usando TCPDUMP

```
17:21:35.483627  IP 0.0.0.0.68 > 255.255.255.255.67:
BOOTP/DHCP, Request from 00:00:00:aa:00:00, length 300

17:21:36.485115  IP 192.0.2.10.67 > 192.0.2.11.68:
BOOTP/DHCP, Reply, length 300
```

Fonte: Elaborado pelo autor.

No exemplo, temos (na linha em negrito), a mensagem de retorno do servidor (no caso, **192.0.2.10**) enviando o pacote Ethernet já com o novo IP da máquina (**192.0.2.11**). Os dois últimos números dos endereços vistos no

TCPDUMP (na linha em negrito temos 67 na origem e 68 no destino) estão relacionados ao protocolo de transporte, que veremos no próximo capítulo.

No lado servidor é necessário instalar o serviço DHCP e configurar o arquivo chamado *dhcpd.conf*, que terá principalmente as informações sobre o número IP usado, a máscara e quais faixas de endereços serão distribuídas. Este arquivo poderá incluir outras informações que podem ser enviadas para a máquina solicitante, como o servidor de DNS, ou o *Default Gateway*.

Quadro 4.6 – Exemplo de arquivo de configuração *dhcpd.conf*

```
subnet 192.0.2.0 netmask 255.255.255.0 {  
    pool {  
        range 192.0.2.10 192.0.2.20;  
    }  
}
```

Fonte: Elaborado pelo autor.

### 4.3.3 DNS (Domain Name System)

Vimos no capítulo 3 as vantagens de utilizar um nome em vez de um endereço numérico e a estrutura já criada para gerenciar e manter este mecanismo em funcionamento. Fica muito mais fácil de lembrar um endereço no formato *yanko@livro.com.br* do que *yanko@192.111.40.5* ainda mais quando qualquer erro com algum dos algarismos pode levar à não conexão.

Mas então, como uma máquina que apenas entende os endereços numéricos do Ethernet ou do IP consegue acessar um domínio como *www.nic.br*?

Na verdade, não consegue: o texto *www.nic.br* precisa ser convertido para um endereço IP e só assim é possível acessar o servidor onde a informação se encontra.

No início, os sistemas operacionais tinham (e ainda possuem) um arquivo simples chamado *hosts*, com duas informações: um endereço IP e um

texto. Cada vez que algum comando precisava acessar outro dispositivo e era digitado um texto em vez do endereço IP (ex: *ping servidor1*), o sistema verificava o arquivo *hosts*, pesquisava se existia o texto e utilizava o IP que estava indicado na mesma linha.

Quadro 4.7 – Exemplo de conteúdo do arquivo /etc/hosts (Linux)

```
YY2 etc # cat hosts
127.0.0.1      localhost
127.0.1.1      YY2
192.168.1.33   www.livroyanko.com.br
```

Fonte: Elaborado pelo autor.

Apesar de ainda existir o arquivo *hosts* (tanto no Linux quanto no Windows), atualmente a conversão entre o domínio e o endereço numérico tem relação com o *protocolo DNS* (*Domain Name System*, ou sistema de nome de domínios). Isso foi necessário pois a quantidade de domínios cresceu muito e ficava impossível manter um arquivo numa máquina local com todas as opções, sem falar no espaço em disco.

### Saiba mais

Veja em seu sistema operacional o conteúdo do arquivo *hosts* e faça testes colocando domínios conhecidos e endereços IPs de sua rede interna ou de outros sites. Não esqueça de retirar depois dos testes, caso contrário não conseguirá mais acessá-los. No Linux, este arquivo se encontra na pasta */etc*. No Windows, este arquivo está na pasta *c:\Windows\System32\Drivers\etc*.

Este protocolo automatiza a consulta dos domínios como se fosse um grande arquivo *hosts* que várias máquinas pudessem consultar.

Para saber como o protocolo DNS funciona, vejamos as ações do sistema operacional passo a passo ao tentar acessar um sistema usando o domínio.

1. O usuário utiliza o domínio diretamente no comando ou aplicativo (ping, browser).

2. O sistema identifica que não é um endereço IP e verifica na sua configuração interna qual o IP de um servidor DNS.
3. Caso não tenha, não é possível acessar e retorna uma mensagem de erro.
4. Se tiver o endereço de um servidor DNS, o sistema cria um pacote com o IP de destino como sendo do DNS e de origem sendo o seu endereço, colocando nos dados o texto que precisa ser transformado.
5. O servidor de DNS recebe o texto e verifica se tem o texto ou repassa (usando as regras de domínio genérico ou de países) para o próximo servidor de DNS que deve conhecer aquele tipo de domínio. Assim sucessivamente até encontrar o servidor DNS final.
6. O servidor que administra o domínio final pesquisa em seus arquivos de configuração qual o IP que está associado aquele nome. Então retorna este IP para a máquina que fez a solicitação.

Apenas após todo este processo é que o comando ou aplicativo sendo utilizado na máquina de origem faz o acesso ao IP de destino que recebeu do servidor de DNS.

Para configurar um servidor DNS temos o sistema BIND, que é um serviço que implementa o protocolo e organiza os arquivos internos com a configuração dos nomes a serem utilizados para cada máquina, a estrutura de acesso a subdomínios e a atualização de um cache interno. Este sistema é composto de registros que funcionam como as linhas de um arquivo *hosts*, com a implementação de algumas rotinas adicionais. No quadro a seguir temos os principais registros do serviço de DNS.

Quadro 4.8 – Lista dos principais tipos de registros de DNS

Registro	Descrição
A	Este registro indica qual o endereço IP estará associado ao nome.
CNAME	Pode redirecionar um nome a outro domínio.
NS	Indica qual o nome do servidor de DNS do atual domínio.

Registro	Descrição
MX	Indica o servidor que irá tratar os e-mails enviados para o domínio.

Fonte: Elaborado pelo autor com base na RFC 1034 (RFC 1034, 2016).

Estes mesmos registros podem ser vistos nas interfaces web de painéis de provedores e inclusive no **Registro.br**, como podemos ver na figura a seguir.

Figura 4.6 – Interface do Registro.br para configuração de domínios e subdomínios



Na interface que vemos na figura 4.6, podemos colocar mais um subdomínio, como por exemplo, www, sistema, mail, ou ww2, e redirecionar para um IP. Assim, quem digitar o endereço **sistema.yrc.com.br** será redirecionado para este novo IP. Cada nome pode ser direcionado para um servidor diferente conforme a necessidade.

Da mesma maneira que os domínios estão sob uma estrutura hierárquica, os servidores de DNS também obedecem este tipo de organização e são distribuídos nos países com esta finalidade. Os servidores que iniciam a estrutura são chamados de *servidores raiz* (*Root Servers*), e podem ser identificados em <http://www.root-servers.org/>.

## 4.4 IPV6

Como vimos também no capítulo 3, o endereçamento IP na versão atual (IPv4) esgotou. Não existem mais faixas de números a serem distribuídos pelo ICANN e os provedores apenas têm ainda alguma reserva que está sendo usada com cautela. De qualquer maneira, não há como ignorar a sua exaustão. Mas, como já temos mostrado em alguns trechos ao longo dos textos, temos uma nova versão do protocolo que está sendo usado aos poucos, conforme a transição vai evoluindo.

É evidente que, considerando o principal problema do IPv4 como o tamanho do endereço (32 bits), este foi o campo que sofreu uma grande alteração. Como vimos anteriormente, o tamanho deste campo saltou de 32 bits para 128 bits, criando uma quantidade muito maior de possibilidades, como podemos verificar no quadro 4.9.

Quadro 4.9 – Número de combinação de endereços por versão

Versão	Combinações (em decimal)
IPv4	4.294.967.296
IPv6	340.282.366.920.938.463.463.374.607.431.768.211.456

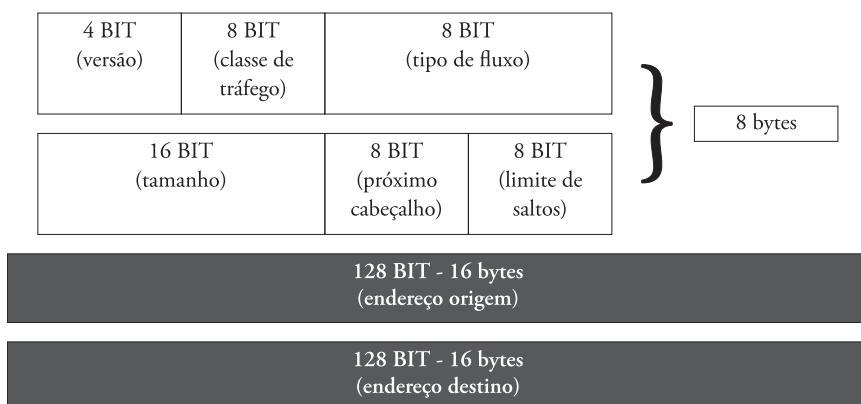
Fonte: Elaborado pelo autor.

Para poder trabalhar com estes endereços de 128 bits, foi alterado o cabeçalho do protocolo IP para acomodar este incremento. Também foi feita higienização nos campos, retirando algumas informações que não eram mais necessárias, redundantes ou que impactavam o desempenho dos roteadores.

#### 4.4.1 Cabeçalho do protocolo IP versão 6

Com essa modificação, foi deixado o cabeçalho com 40 bytes fixos e a possibilidade de acrescentar alguns trechos em caso de extensão. Na prática, a maior parte dos pacotes pode ser trabalhada pelos roteadores com o cabeçalho padrão, deixando o desempenho do processo de avaliação dos pacotes mais simples.

Figura 4.7 – Cabeçalho do protocolo IPv6



Na figura 4.7, temos o cabeçalho do IPv6 iniciando com a versão do protocolo, neste caso 6 (0110). Com a continuidade da versão no início, os dois tipos de protocolos podem coexistir, pois o roteador verifica primeiro a versão nos 4 primeiros bits e, conforme for, utiliza o algoritmo adequado àquele momento.

Depois, temos a identificação do *tipo de tráfego* para a implementação de prioridades de encaminhamento e em seguida temos o *tipo de fluxo* para agrupar os tipos de pacotes com o mesmo tipo de necessidade. O campo do *tamanho* agora indica apenas a parte dos dados (o cabeçalho não entra na contagem) e o *próximo cabeçalho* indica se existe um cabeçalho de extensão ou, se não tiver, qual o próximo protocolo na camada de transporte. Por fim, o limite de saltos trabalha como o campo TTL (*time to live*) usado no IPv4, decrementando a cada roteador no meio do caminho. Na parte dos endereços, apenas o aumento do campo para 128 bits.

#### 4.4.2 Endereços

Além da quantidade de combinações, a notação usada para indicar o endereço foi alterada. Agora temos o caractere ":" como separador, os números são indicados na base Hexadecimal (como os *mac address*) e estão agrupados em 8 grupos de 4 dígitos hexa. A cada dois dígitos hexa temos a representação de 8 bits. Desta forma, cada grupo representa 16 bits (8 grupos de 16 bits = 128 bits). Como exemplo temos o endereço **2001:12ff:0:4:0:0:6** do domínio *www.nic.br*.

#### Saiba mais

O comitê gestor de Internet no Brasil mantém um site específico sobre IPv6 ([www.ipv6.br](http://www.ipv6.br)), que pode ser utilizado para se aprofundar sobre este novo protocolo. Possui também tutoriais, vídeos e eventos sobre o assunto.

Para facilitar a digitação e leitura, foram feitas algumas regras com relação à notação, abreviando os endereços. Uma delas indica que, quando tiver um ou mais grupos seguidos compostos somente com zeros, eles podem ser

omitidos. Mas deve ser observado que, caso existam dois grupos de zeros intercalados por números hexa, a omissão dos zeros deve ocorrer apenas em um dos grupos. Usando como exemplo o endereço do nic.br acima, ele pode ser reescrito como: **2001:12ff:0:4::6**. O endereço de *loopback* que normalmente seria escrito como **0:0:0:0:0:1** fica como **::1**. Também não é necessário indicar o zero quando este está à esquerda dos números.

Os utilitários de rede precisaram ser modificados para trabalhar com o endereço maior. Vejamos a seguir o resultado do **ping6**, que é a versão do **ping** para trabalhar com IPv6, fazendo um teste na interface de *loopback*.

Quadro 4.10 – Resultado ping6 (GNU/Linux)

```
YY2 etc # ping6 -c4 ::1
PING ::1(::1) 56 data bytes
64 bytes from ::1: icmp_seq=1 ttl=64 time=0.023 ms
64 bytes from ::1: icmp_seq=2 ttl=64 time=0.052 ms
64 bytes from ::1: icmp_seq=3 ttl=64 time=0.044 ms
64 bytes from ::1: icmp_seq=4 ttl=64 time=0.087 ms
--- ::1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.023/0.051/0.087/0.024 ms
```

Fonte: Elaborado pelo autor.

Grandes sites e provedores já estão utilizando o novo protocolo, mas ainda temos o IPv4 muito forte nas redes de empresas, além de alguns softwares ainda não terem sido modificados para comportar o novo endereçamento. Mas a transição é inevitável, uma vez que os endereços IPv4 já se esgotaram.

## 4.5 Utilitários

Os utilitários a seguir são utilizados no diagnóstico de problemas nas redes, mas também permitem testar e praticar os conceitos obtidos até agora. Eles permitem avaliar os caminhos feitos pelos pacotes de rede e avaliar a situação do DNS.

Quando usados em conjunto com os outros utilitários já demonstrados nos capítulos anteriores e com as ferramentas de simulação como o CORE, eles permitem o entendimento melhor de como a rede trabalha.

#### 4.5.1 Traceroute

O comando *traceroute* (ou *tracert*, no Windows), permite identificar cada roteador no caminho do pacote até o seu destino. Pode ser usado em redes internas para identificar o ponto de saída da rede local, ou pode ser usado para descobrir o caminho do pacote na Internet.

No quadro 4.11 temos dois exemplos de retorno do *traceroute* sendo usado para descobrir o caminho utilizado, desde o computador de origem até um dos servidores de DNS do Brasil. O primeiro exemplo na linha de cima da tabela, mostra todos os roteadores no caminho com o seu nome de domínio identificado a cada ponto.

Quadro 4.11 – Exemplos de *traceroute*

```
YY2 # traceroute a.dns.br

traceroute to a.dns.br (200.160.0.10), 30 hops max, 60 byte packets
1 192.168.0.1 (192.168.0.1)  2.262 ms 2.634 ms 3.570 ms
2 10.50.64.1 (10.50.64.1)  14.019 ms 14.364 ms 14.369 ms
3 bd040072.virtua.com.br (189.4.0.114)  15.269 ms 15.592 ms 20.109
ms
4 200.219.140.103 (200.219.140.103)  26.170 ms 26.736 ms 26.670 ms
5 et-4-0-0-0ptx-a.spo511.algaratelecom.com.br      (170.84.34.70)
26.759 ms 26.714 ms 26.794 ms
6 et-4-2-0-0.edge-c.spo511.algaratelecom.com.br      (170.84.33.73)
26.748 ms 15.433 ms 16.214 ms
7 xe-0-0-2.537.gw1.nu.registro.br (187.32.53.69)  19.020 ms 19.386
ms 19.326 ms
8 xe-5-0-1-0.core1.nu.registro.br (200.160.0.166)  22.092 ms
22.039 ms 21.596 ms
9 xe-0-0-0.gw.a.dns.br (200.160.0.246) 21.999 ms 24.767 ms 24.298
ms
10 a.dns.br (200.160.0.10)  23.391 ms 23.474 ms 25.913 ms
```

```
YY2 # traceroute -n a.dns.br
traceroute to a.dns.br (200.160.0.10), 30 hops max, 60 byte packets
1 192.168.0.1 55.981 ms 68.447 ms 68.405 ms
2 10.50.64.1 68.375 ms 68.349 ms 68.324 ms
3 189.4.0.114 62.175 ms 62.301 ms 62.286 ms
4 200.219.140.103 68.179 ms 68.142 ms 68.080 ms
5 170.84.34.70 68.055 ms 68.031 ms 68.015 ms
6 170.84.33.73 67.974 ms 64.222 ms 74.199 ms
7 187.32.53.69 74.080 ms 74.006 ms 68.389 ms
8 200.160.0.166 68.374 ms 68.365 ms 68.165 ms
9 200.160.0.246 70.626 ms 70.646 ms 70.647 ms
10 200.160.0.10 65.573 ms 67.231 ms 68.256 ms
```

Fonte: Elaborado pelo autor.

Na segunda linha, temos o comando *traceroute* com a opção (-n) informada. Esta opção indica que o comando não vai verificar o DNS de cada endereço de roteador encontrado. Esta opção deixa o processamento do *traceroute* bem mais rápido. Nos dois exemplos, temos em destaque o último roteador que liga o servidor de destino.

A partir de cada computador que esteja conectado em provedores diferentes, pode ser visto um caminho diferente do mostrado. Quando usado com a pesquisa de DNS habilitada, com base nos domínios dos roteadores, muitas vezes é possível ver quais os provedores utilizados e como estão interligados. Considerando que o *traceroute* normal utiliza o ICMP como retorno das tentativas, nos casos em que esse protocolo é filtrado nas redes, podem ser utilizadas outras variações de pesquisa, como o uso de UDP e TCP (estes dois protocolos serão vistos no próximo capítulo).

#### 4.5.2 MTR

O *mtr* é um comando que usa o mesmo princípio do *traceroute* e do *ping*, mas mantém a tela atualizada com os tempos de resposta de cada roteador. Com isso, podemos avaliar a situação de cada trecho ao longo do tempo.

Veja no quadro a seguir dois exemplos do uso do *mtr* ao avaliar o caminho até o servidor *a.dns.br*.

Quadro 4.12 – Resultados do *mrt*

My traceroute [v0.85]							Fri Sep 30 00:28:08 2016							
YY2 (0.0.0.0)	Keys:	Help	Display mode	Restart	statistics	Order of fields	quit	Packets	Pings					
Host								Loss%	Snt	Last	Avg	Best	Wrst	StDev
1. 192.168.0.1								0.0%	98	1.0	1.9	0.9	49.0	4.9
2. 10.50.64.1								0.0%	98	11.2	11.9	7.5	187.8	18.0
3. 189.4.0.114								0.0%	98	11.9	12.7	8.5	42.1	6.5
4. 200.219.140.103								0.0%	98	17.6	16.8	14.5	54.8	4.2
5. 170.84.34.70								0.0%	98	40.6	17.2	10.7	42.9	4.2
6. 170.84.33.73								0.0%	98	16.5	17.3	14.2	61.8	5.1
7. 187.32.53.69								0.0%	97	15.8	17.4	12.4	78.6	7.1
8. 200.160.0.166								0.0%	97	16.0	18.4	12.3	66.8	6.2
9. 200.160.0.246								0.0%	97	12.0	18.1	12.0	59.1	5.1
10. 200.160.0.10								0.0%	97	16.3	17.2	12.4	39.7	3.9

My traceroute [v0.85]							Fri Sep 30 00:26:44 2016							
YY2 (0.0.0.0)	Keys:	Help	Display mode	Restart	statistics	Order of fields	quit	Packets	Pings					
Host								Loss%	Snt	Last	Avg	Best	Wrst	StDev
1. 192.168.0.1								0.0%	98	1.0	1.9	0.9	49.0	4.9
2. 10.50.64.1								0.0%	98	11.2	11.9	7.5	187.8	18.0
3. bd040072.virtua.com.br								0.0%	98	11.9	12.7	8.5	42.1	6.5
4. 200.219.140.103								0.0%	98	17.6	16.8	14.5	54.8	4.2
5. et-4-0-0-optx-a.spo511.algaratele								0.0%	98	40.6	17.2	10.7	42.9	4.2
6. et-4-2-0-0.edge-c.spo511.algarate								0.0%	98	16.5	17.3	14.2	61.8	5.1
7. xe-0-0-2.537.gwl.nu.registro.br								0.0%	97	15.8	17.4	12.4	78.6	7.1
8. xe-5-0-1-0.corel.nu.registro.br								0.0%	97	16.0	18.4	12.3	66.8	6.2
9. xe-0-0-0.gw.a.dns.br								0.0%	97	12.0	18.1	12.0	59.1	5.1
10. a.dns.br								0.0%	97	16.3	17.2	12.4	39.7	3.9

Fonte: Elaborado pelo autor.

Como podemos ver acima, de maneira semelhante ao *traceroute*, pode-se alternar para a verificação ou não do domínio dos roteadores. Mas as opções são dinâmicas: ao clicar “n” no teclado temos a troca da opção.

### 4.5.3 DIG

O *dig* é um utilitário que permite investigar a situação dos registros de DNS de um domínio. Na sua forma mais simples, o comando *dig domínio* mostra as opções de registro de endereço IP (registro A). No exemplo a seguir

temos o resultado do comando **dig fael.edu.br** com o retorno dos endereços IPs sendo usados pelos servidores de DNS e do servidor principal do domínio.

Quadro 4.13 – Resultado do comando DIG

```
1. ; <>> Dig 9.9.5-3ubuntu0.8-Ubuntu <>> fael.edu.br
2. ;; global options: +cmd
3. ;; Got answer:
4. ;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 52345
5. ;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 3
6.
7. ;; OPT PSEUDOSECTION:
8. ;; EDNS: version: 0, flags:; udp: 4096
9. ;; QUESTION SECTION:
10. ;fael.edu.br.      IN  A
11.
12. ;; ANSWER SECTION:
13. fael.edu.br. 561800 IN A 54.83.97.8
14.
15. ;; AUTHORITY SECTION:
16. fael.edu.br. 85894 IN NS ns2.fael.edu.br.
17. fael.edu.br. 85894 IN NS ns1.fael.edu.br.
18.
19. ;; ADDITIONAL SECTION:
20. ns1.fael.edu.br. 85894 IN A 54.87.239.128
21. ns2.fael.edu.br. 85894 IN A 52.44.51.70
22.
23. ;; Query time: 10 msec
24. ;; SERVER: 127.0.1.1#53(127.0.1.1)
25. ;; WHEN: Sun Oct 02 11:19:32 BRT 2016
26. ;; MSG SIZE rcvd: 124
```

Fonte: Elaborado pelo autor.

No quadro 4.13 os resultados efetivos da consulta sobre o domínio *fael.edu.br* estão em negrito. As demais linhas são observações do programa. Na linha 13, temos o IP do servidor do domínio (**54.83.97.8**). Nas linhas 16 e 17 temos os servidores de DNS deste domínio e nas linhas 20 e 21 vemos qual o IP dos dois servidores de DNS. No fim da consulta temos o tempo que demorou a consulta e o IP do servidor que foi usado para fazer a consulta de DNS. Neste caso, foi usado o endereço local da máquina, que depois repassou para o servidor de DNS do provedor.

Podemos também indicar qual servidor queremos usar para a consulta, assim como fazer consulta de outros registros. No exemplo a seguir, está um resumo do resultado (sem os comentários) de uma consulta feita com um servidor específico de DNS e a lista dos servidores de e-mail do domínio.

Quadro 4.14 – Resultado do comando DIG (*dig @8.8.8.8 fael.edu.br MX*)

```

1. ;; ANSWER SECTION:
2. fael.edu.br.      21599 IN  MX    1 aspmx.l.google.com.
3. fael.edu.br.      21599 IN  MX    5 alt1.aspmx.l.google.com.
4. fael.edu.br.      21599 IN  MX    10 aspmx3.googlemail.com.
5. fael.edu.br.      21599 IN  MX    5 alt2.aspmx.l.google.com.
6. fael.edu.br.      21599 IN  MX    10 aspmx2.googlemail.com.

```

Fonte: Elaborado pelo autor.

Neste último exemplo, foi usado um servidor de DNS externo (servidor gratuito do Google – 8.8.8.8), e foi solicitada a lista de todos os servidores que respondem por e-mail (MX) deste domínio (linhas 2 a 6).

Utilizando a opção “-x” e indicando um IP, o DIG pode fazer a função inversa: pesquisar qual nome de domínio está associado ao IP.

#### 4.5.4 Whois

O *whois* é um utilitário que consulta os registros da base de dados de domínios para identificar qual o responsável pelo domínio pesquisado.

Quadro 4.15 – Consulta do domínio *nic.br*

```

YY2 - # whois nic.br

% Copyright (c) Nic.br
% The use of the data below is only permitted as described in
% full by the terms of use at http://registro.br/termo/en.html ,
% being prohibited its distribution, commercialization or
% reproduction, in particular, to use it for advertising or
% any similar purpose.
% 2016-10-02 13:30:21 (BRT -03:00)

domain:      nic.br
owner:       Nucleo de Inf. e Coord. do Ponto BR - NIC_BR
ownerid:     005.506.560/0001-36
responsible: Demi Getschko
country:     BR
owner-c:     FAN
admin-c:     FAN
tech-c:      FAN
billing-c:   FAN
nserver:    a.dns.br
nsstat:     20160927 AA
nslastaa:   20160927
nserver:    b.dns.br
nsstat:     20160927 AA

```

```
nslastaa: 20160927
nserver: d.dns.br
nsstat: 20160927 AA
nslastaa: 20160927
nserver: e.dns.br
nsstat: 20160927 AA
nslastaa: 20160927
dsrecord: 57436 RSA/SHA-1 CCB7D717A8868B8739A78FEC8FB60E62EBE2D89B
dsstatus: 20160927 DSOK
dslastok: 20160927
created: 19970711 #46903
changed: 20070606
status: published
```

Fonte: Elaborado pelo autor.

Pode-se testar os domínios e verificar quando foram criados, se estão ativos, quais os servidores de DNS estão associados a ele e a quem pertence o domínio (CNPJ ou CPF).

## Síntese

Vimos neste quarto capítulo o detalhamento da camada de rede da pilha de protocolos TCP/IP e um aprofundamento na forma como o IP é estruturado. Pudemos avaliar cada um dos campos que fazem parte do cabeçalho do protocolo, tanto na versão 4 quanto na nova versão, a 6. Os utilitários apresentados podem auxiliar na compreensão de como funciona o protocolo IP, mas também vão auxiliar a detectar problemas na rede e na conexão da rede na Internet. Vimos neste capítulo as notações usadas para os endereços IPs e como podemos configurar subredes usando as máscaras. Alguns mecanismos para conversão de endereços, como o NAT e o ARP foram apresentados, assim como podem ser feitas as configurações relacionadas aos domínios na internet.

E com o DHCP, pudemos verificar uma forma de distribuir e configurar os endereços nas máquinas automaticamente, de maneira a impedir o uso de um endereço IP por mais de uma máquina na rede.

## Atividades

1. Marque com V (verdadeiro) e F (falso) as afirmações sobre a camada de rede estudadas nesse capítulo:
  - ( ) o roteador faz a conversão de domínio para IP antes de enviar o pacote pela interface de rede.
  - ( ) a versão 4 do protocolo IP foi criada com endereços de 32 bits separados em 4 números de 0 a 255.
  - ( ) os IPs privados da versão 4 do protocolo foram as faixas vendidas para as empresas que queriam se conectar na internet.
  - ( ) o registro MX do servidor de DNS indica o endereço do servidor que será responsável pelos e-mails do domínio.
2. Um administrador de rede deixou documentado que a rede da empresa iria usar o endereço 192.168.100.0/23. Calcule a quantidade de endereços que poderão ser usados nas máquinas.
3. Por que o uso do NAT (Network Address Translation) conseguiu adiar o esgotamento dos endereços IP públicos?
4. Caso uma máquina não tenha um endereço IP configurado na sua interface de rede, como é o processo de solicitação automática de endereço?



# 5

## Camada de transporte

NESTE CAPÍTULO, DISCUTIREMOS a camada de transporte do modelo TCP/IP, cujos protocolos criam um fluxo de dados diretamente entre dois dispositivos, permitindo que os serviços dos dois pontos da comunicação possam ser identificados e consigam trocar as informações.

O PROTOCOLO TCP (*Transmition Control Protocol*), que veremos com detalhes adiante, permite o controle de como os pacotes de dados são gerenciados durante a conexão. Caso haja uma interrupção, erro ou inversão de pacotes na chegada, a camada de transporte tem a missão de verificar a situação e prover um mecanismo para que, se for necessário, o pacote ou a sequência de pacotes seja reenviado.

Essa camada de transporte também tem outro protocolo, o chamado UDP (*User Datagram Protocol*), que é mais voltado ao desempenho e não executa as verificações de retransmissão ou organização de fluxo, sendo utilizado em alguns casos específicos.

## 5.1 Endereços (portas)

Já foi exposto em diversos trechos ao longo dos capítulos que um dispositivo pode se conectar com vários outros na rede. Quando isso acontece, o IP de origem é o mesmo, e o de destino é diferente para cada dispositivo a ser conectado. Mas há conexões que acontecem entre dois dispositivos envolvendo mais de um serviço no mesmo equipamento. Por exemplo, um servidor que atende por e-mail e também hospeda um site: ele terá um IP, mas cada serviço tem um formato de comunicação diferente. Nessa situação, um outro dispositivo tentará se conectar ao serviço de site, mas a única informação que teria para isso seria o IP de origem (dele próprio) e o IP de destino (do servidor). Ao chegar a solicitação no servidor, por meio apenas dessas informações do endereço IP não seria possível identificar qual ou se existe o serviço a ser conectado.

Para se conseguir separar e identificar os serviços disponíveis e orientar a comunicação por meio dos dois pontos, temos um código próprio da camada de transporte: a porta, que é como se fosse a numeração da sala de um prédio comercial. O IP seria o número do prédio na rua, e o código da porta, a sala onde se terá a prestação de serviço (dentista, advogado). Da mesma forma que em um prédio existem várias empresas, em um IP posso ter vários serviços de comunicação, como veremos ao detalhar o estudo sobre o TCP e UDP. Por ser um número de 16 bits, o total de portas possíveis a cada IP configurado na interface de um dispositivo é de 65.536.

Quando se tem um conjunto de IP de origem, IP de destino, porta de origem e porta de destino, temos o que chamamos de *socket*, que permite diferenciar uma conexão das outras, mesmo que sejam feitas pelos mesmos dispositivos em cada ponta. No quadro 5.1, temos algumas conexões de exemplo entre um cliente e um dispositivo de destino, em que estão apresentadas as combinações de endereços de IP de origem e porta de origem, assim como os IP's de destino e portas de destino. Em todas as combinações a seguir, mesmo

quando o dispositivo de origem acessa o mesmo serviço remoto, temos a diferenciação da conexão pela porta de origem diferente (linhas 5 e 6 do quadro).

Quadro 5.1 – Conexões entre dispositivos

	IP Origem	Porta Origem	IP Destino	Porta Destino
<b>1</b>	192.168.1.4	50000	192.168.1.12	80
<b>2</b>	192.168.1.4	50020	192.168.1.5	25
<b>3</b>	192.168.1.33	50000	192.168.1.12	80
<b>4</b>	192.168.1.33	50040	192.168.1.5	25
<b>5</b>	192.168.1.33	50060	192.168.1.5	80
<b>6</b>	192.168.1.33	50070	192.168.1.5	80

Fonte: Elaborado pelo autor.

Um dispositivo pode oferecer apenas um endereço de porta de cada vez. Caso tenha mais de um serviço a ser disponibilizado, deve utilizar um endereço de porta diferente para evitar o conflito. Um sistema operacional verifica e emite um alerta ao tentar usar um endereço de porta já em uso por outro serviço no momento de execução.

### 5.1.1 Testando as conexões usando NCAT

Para visualizar e testar as conexões por meio de portas e IP's, usaremos, ao longo deste capítulo, o utilitário *ncat*, que permite associar uma porta a uma conexão e também fazer uma conexão a um IP e porta de destino específicos.

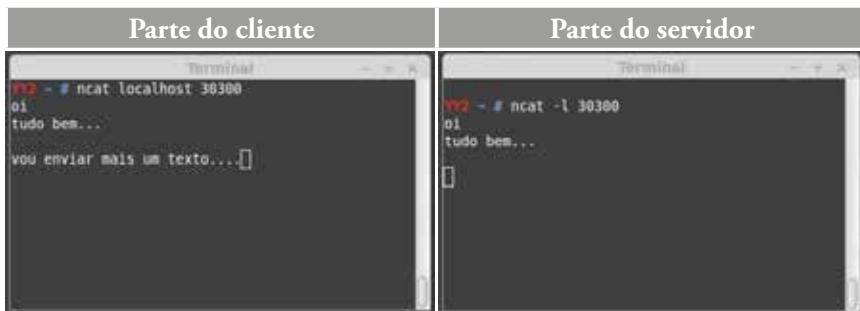
#### Saiba mais

O utilitário *ncat* original foi disponibilizado em 1995, mas foi abandonado pelo autor após alguns anos. Devido à popularidade do utilitário e sua flexibilidade, foram feitas várias versões do comando. Utilizaremos a versão mantida pelo grupo mantenedor do projeto NMAP (<https://nmap.org/ncat/>), que pode ser instalada via linha de comando no Linux Mint por meio do “*apt-get install nmap*”, que instala o “*nmap*”, “*ncat*” e “*nping*”.

Para simularmos a disponibilização de um serviço na rede, podemos usar o *ncat* com a opção “-l NUM-PORTA”. Assim, ele pode ficar “escutando” se ocorre alguma tentativa de acesso e, caso seja o número da porta indicada, ele aceita a conexão. O quadro 5.2 mostra a utilização do *ncat* nas duas funções básicas: conectar e disponibilizar uma conexão.

Quando um dispositivo oferece um serviço em uma porta, ele é chamado de servidor e, quando é feito o acesso do serviço por outro dispositivo, chamamos de cliente. Assim, temos a arquitetura cliente-servidor, que é quando o sistema proposto tem uma parte que disponibiliza um serviço e outra, que consome esse serviço de forma remota.

Quadro 5.2 – Conexão via NCAT



Fonte: Elaborado pelo autor.

No quadro anterior, no lado do servidor temos a disponibilização de um acesso via porta **30300**. O *ncat* ficará aguardando até que seja feita uma conexão a essa porta. Na parte do cliente, temos o acesso a um dispositivo chamado *localhost* e na porta 30300. Essa simulação pode ser feita na mesma máquina com duas janelas fazendo o acesso ao endereço IP de teste *loopback* (usando o nome de máquina padrão *localhost*). Apesar de serem apenas duas janelas na mesma máquina, o acesso é feito por meio do IP 127.0.0.1 e da porta TCP 30300.

No lado “Parte do cliente”, foram digitadas algumas palavras e finalizou-se com <ENTER>. Em cada caso, ao apertar o <ENTER> a mensagem era enviada para o outro *ncat* (lado “Parte do servidor”), que nada mais faz do que imprimir o que foi recebido. Nesse caso, temos na “Parte do servidor” as

mensagens que já foram recebidas e impressas na tela: “*oi*” e “*tudo bem...*”. Na “Parte do cliente”, temos, ainda, uma frase, “*vou enviar mais um texto*”, que não foi finalizada com o <ENTER> e, com isso, ainda não aparece na “Parte do servidor”.

Sempre que se inicia uma conexão o sistema operacional escolhe aleatoriamente uma porta de origem que ainda não esteja sendo usada.

Com o *ncat* pode-se, também, testar o que acontece ao tentar disponibilizar uma porta já em uso. No quadro a seguir, considerando que ainda exista a mesma porta utilizada no exemplo anterior, disponibilizada pelo outro *ncat*, ao tentar executar o comando reservando a mesma porta (30300), o sistema operacional não vai liberar e retornará uma mensagem com a situação: endereço já está em uso (*address already in use*).

Quadro 5.3 – Tentativa de uso da mesma porta

```
YY2 ~ # ncat 127.0.0.1 -l 30300
Ncat: bind to 127.0.0.1:30300: Address already in
use. QUITTING.
```

Fonte: Elaborado pelo autor.

Quando um aplicativo está usando uma porta e é instalado outro aplicativo que precisa disponibilizar a conexão na mesma porta, deve-se ou acrescentar mais um endereço IP ou alterar a porta usada na disponibilização do serviço.

### 5.1.2 Serviços X portas

No exemplo de conexão entre os *ncat* (quadro 5.2), foi acessado um serviço na porta 30300 que serviu apenas para enviar caracteres até o servidor após o <ENTER>.

Mas como foi identificado que era no endereço de porta **30300** que o outro *ncat* estava aguardando a conexão? E se o serviço disponibilizado estivesse em um servidor em outro país, como poderia ser descoberto em qual porta ele estaria sendo divulgado?

Pelo motivo de não se saber antecipadamente em qual porta cada serviço pode ser acessado, convencionou-se que algumas portas estariam reser-

vadas para um tipo de aplicação. Essa convenção facilita encontrar o recurso que está sendo compartilhado, bastando para isso saber o IP (ou domínio na Internet, como foi visto em capítulo anterior) e acrescentar a porta padrão para aquele tipo de compartilhamento.

As portas estão normalmente associadas a protocolos da camada de aplicação, que veremos no próximo capítulo.

Quadro 5.4 – Exemplos de algumas portas e seus serviços associados

Porta	Serviço Padrão
21	Aplicação relacionada à cópia de arquivos entre servidores usando o protocolo FTP.
22	Porta usada pelo protocolo SSH que permite o uso de linha de comando remota, com proteção de criptografia.
53	Endereço de porta do serviço de conversão de DNS.
67	Porta utilizada pelo protocolo DHCP para identificar a solicitação, um pedido de endereço IP por um dispositivo quando é iniciado. Vista no capítulo 4.
80	Aplicação que permite a conexão a sites usando o servidor WEB, por meio do protocolo HTTP.
161	Porta usada pelo protocolo SNMP, que coleta informações remotamente dos dispositivos da rede. Visto no capítulo 4, em gerenciamento de redes.
445	Usada pelos sistemas Windows para compartilhamento de arquivos em uma rede Microsoft, utilizando o protocolo CIFS.
1433	Porta associada ao banco de dados Microsoft SQL Server.
3306	Porta associada ao banco de dados MySQL.

Fonte: Elaborado pelo autor.

Cada uma das portas descritas anteriormente pode ser utilizada em outros serviços ou protocolos, mas para facilitar a divulgação e o uso dos serviços são normalmente associadas aos protocolos acima. Conforme RFC6335 (2016), a sequência de portas de 0 a 1023 é conhecida como portas de sistema e, na maior parte dos sistemas operacionais, é preciso ter direitos de administrador

para usar alguma delas. De 1024 a 49151 são portas que podem ser registradas para uso, mas não requerem privilégios para serem associadas a serviços nos dispositivos.

No GNU/Linux, no arquivo “/etc/services”, pode-se listar quais os nomes de serviços associados ao número de portas que o sistema operacional mostrará caso precise identificar a porta com um nome. Mas isso poderá acarretar confusão caso a porta esteja sendo utilizada por um aplicativo que usa um protocolo diferente do padrão. Nesse caso, a informação pode ser interpretada erroneamente. No Windows, também tem-se o mesmo tipo de arquivo, nesse caso localizado em “C:\Windows\System32\drivers\etc\services”.

Ao iniciar uma conexão, o sistema operacional associa um número temporário como porta de origem, normalmente dentro da faixa sugerida pela IANA (2016) – entre 49152 e 65535, mas o sistema operacional pode implementar um conjunto diferente.

O usuário comum não tem muito contato com os endereços de portas. Muitos aplicativos já são configurados para o uso delas. Mesmo nos *browsers* usados diretamente pelos usuários, quando é digitado o endereço de site que se quer acessar, o software anexa automaticamente a porta 80 padrão de serviços WEB ao endereço IP de destino.

Apenas quando o site está usando uma porta diferente da padrão é que o usuário precisa intervir e indicar explicitamente o número da porta separando os dois endereços com dois-pontos (:), por exemplo: 192.168.5.10:**8080**.

## 5.2 TCP (Transmission Control Protocol)

O protocolo TCP organiza um fluxo de dados contínuo entre dois pontos. Para isso, ele deve registrar quando a conexão inicia e, enquanto ela continua acontecendo, mantém a sequência dos pacotes; caso perca algum pelo caminho, aciona um mecanismo para a retransmissão. Esse processo indica que o protocolo é orientado pela conexão.

---

O contexto da conexão orientada do protocolo TCP pode ser comparado com a atividade desenvolvida dentro de uma

estação de ônibus. Considere um grande grupo de pessoas que precisa ir a Joinville para um festival. Dentro da estação de ônibus temos várias linhas que fazem trajetos para várias cidades. Na linha Curitiba-Joinville, cada ônibus estaciona em uma plataforma já identificada, e os passageiros entram nele até preencher todos os assentos. O ônibus segue o trajeto até Joinville e, ao chegar na estação de seu destino, estaciona na plataforma de desembarque, e os passageiros descem. Na estação de Curitiba, cada passageiro restante do grande grupo aguarda um novo ônibus da linha Curitiba-Joinville e ao entrar no transporte já sabe que será levado à estação correta. Mesmo que outros ônibus estejam entrando e saindo da estação de Curitiba, cada identificação da linha faz com que o passageiro já perceba qual é o ônibus correto.

No protocolo TCP é criada uma “linha” de comunicação entre dois dispositivos, mostrando uma identificação daquele trajeto. Os dados são inseridos nos pacotes e trafegam identificados até o seu destino. Cada estação é um endereço IP de origem e chegada, e as plataformas são as portas de origem e chegada do exemplo acima. Assim, cada grupo de dados que sai de um dispositivo, mesmo que sejam separados em grupos menores, consegue ser identificado e reagrupado de maneira correta no seu destino.

---

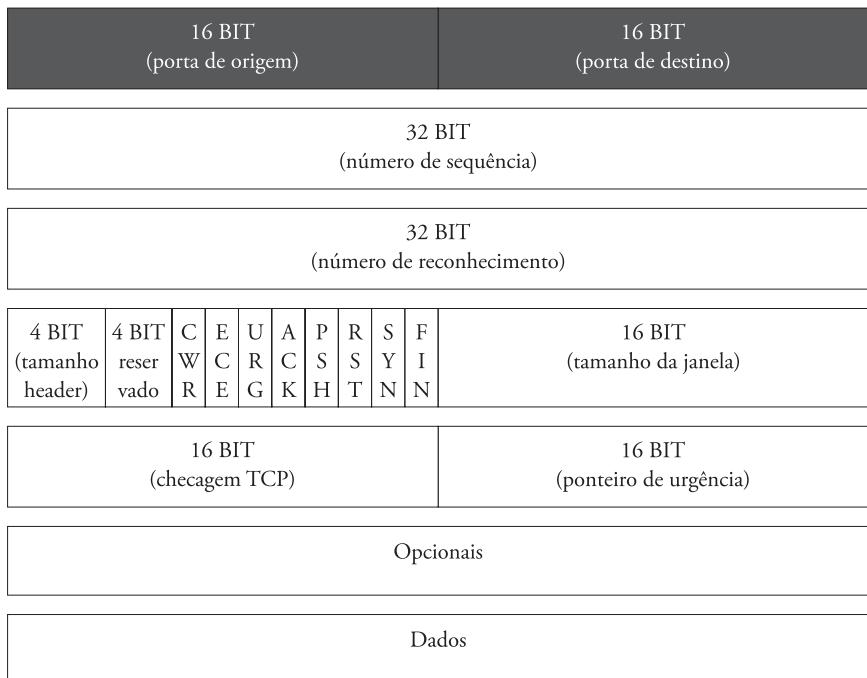
Para perceber que houve uma perda de pacote pelo caminho, o TCP deve manter por um tempo a espera pelo pacote. Quando o tempo finaliza, deve perceber que o pacote não chegará e envia uma requisição de novo pacote.

Para controlar essas e demais funções, o cabeçalho do protocolo TCP possui vários campos que serão detalhados a seguir.

### 5.2.1 Cabeçalho TCP

O cabeçalho TCP apresentado a seguir indica a parte fixa representada pelos campos com a indicação dos bits utilizados, seguido pela parte opcional do protocolo e, no final, a sequência de dados a serem enviados. A parte fixa ocupa um total de 20 bytes.

Figura 5.1 – Cabeçalho TCP



Fonte: Elaborado pelo autor.

No início do cabeçalho do TCP, temos dois conjuntos de 16 bits para os endereços das portas de origem e de destino do protocolo. Cada conjunto pode gerar um endereço de porta entre 0 e 65.535. Logo em seguida, gera um número de sequência de 32 bits, que permitirá identificar um fluxo de conexão. Depois, gera um campo também de 32 bits, que permite estabelecer o reconhecimento do fluxo da conexão (essa mecânica será vista mais adiante quando será detalhado o *HandShake*).

O tamanho do cabeçalho é identificado por um campo de 4 bits (decimal de 0 a 15), que indicam quantos grupos de 32 bits (4 bytes) fazem parte do total do cabeçalho, podendo chegar a um máximo de 60 bytes ( $15 \times 4$  bytes = 60).

Após 4 bits de reserva não usados, seguem 8 bits, que vão identificar a situação do pacote em cada momento do fluxo do TCP. Esses *flags*, quando ligados, indicam: (CRW) e (ECE) – que a rede está sobrecarregada e é necessário reavaliar a conexão; (URG) – que o ponteiro de urgência está ligado; (ACK) – que o número de reconhecimento é válido; (PSH) – que os dados incluídos devem ser liberados diretamente para a aplicação sem passar pelo armazenamento temporário (*buffer*); (RST) – que a conexão deve ser reiniciada; (SYN) – que a conexão será iniciada; (FIN) – que a conexão será finalizada.

O tamanho da janela indica quantos bytes o receptor da comunicação está preparado para receber. Esse número pode variar dentro da transmissão, indicando a “dosagem” de bytes que podem ser recebidos, principalmente relacionados ao tamanho do espaço em sua fila de receção (*buffer*).

Na checagem de TCP, é feito o cálculo para identificar erros na transmissão. Nesse caso, o cálculo inclui os bytes do cabeçalho e do conjunto de dados.

Caso o *flag* URG esteja ligado, o conteúdo do campo do ponteiro de urgência é verificado. Esse mecanismo é usado quando se que precisa fazer uma interrupção do fluxo ou quando a aplicação precisa enviar um sinal de alerta no meio da comunicação para o servidor. Esse mecanismo acabou em desuso.

Depois da parte fixa do cabeçalho, podem ser enviados parâmetros opcionais, mas com tamanho máximo de 40 bytes.

### 5.2.2 Acordo de conexão (HandShake)

Para estabelecer uma comunicação entre dois dispositivos, o TCP cria uma sequência de passos que permite alinhar seus controles e identificadores para que os dois lados da comunicação possam garantir que os dados estão sendo entregues e na ordem correta.

Esse acordo inicial de conexão, conhecido como *HandShake* (aperto de mãos), cria uma sequência de requisições e confirmações que são monitoradas

pelo uso de números de 32 bits que são incrementados à medida que os bytes são transferidos. Cada pacote tem um identificador de início e fim, e a transmissão é cadenciada pelo espaço usado pelo armazenamento temporário dos pacotes no lado receptor com o uso do campo de tamanho da janela (*window*).

Para poder acompanhar e visualizar cada um desses passos, utilizaremos os utilitários *ncat* e *tcpdump* já vistos anteriormente.

Nesse cenário prático, o *ncat* servirá tanto como emissor quanto como receptor da comunicação por meio da interface de teste (*loopback*), e o *tcpdump* mostrará os pacotes sendo trocados entre o emissor e receptor para estabelecer a conexão.

Para iniciar o teste no lado servidor, o *ncat* usa o mesmo processo visto anteriormente: “*ncat -l 30300*”. No lado cliente, é feito o acesso usando: “*ncat localhost 30300*”.

Quadro 5.5 – Conexão TCP usando NCAT

LADO CLIENTE	LADO SERVIDOR
<code>YY2 ~ # ncat localhost 30300</code> teste 123456789 ^C	<code>YY2 ~ # ncat -l 30300</code> teste 123456789

Fonte: Elaborado pelo autor.

No quadro anterior, o cliente acessou o servidor na porta 30300 e enviou primeiro a palavra “*teste*” com <ENTER>, depois, a sequência de 9 algarismos e <ENTER>. Em seguida, interrompeu a comunicação com CTRL+C. No lado servidor, foram recebidas e mostradas na tela as duas sequências enviadas.

Toda a comunicação foi gravada usando o comando “*tcpdump -i lo port 30300 -w hand.pcap*” e salva no arquivo “*hand.pcap*” indicado nas opções. No comando *tcpdump*, também foram filtradas apenas as comunicações da interface *loopback* que estivessem relacionadas à porta escolhida com o filtro “*port 30300*”. O formato do arquivo *tcpdump* é binário e só podem ser apresentadas as informações gravadas usando o próprio comando ou outra ferramenta que conheça esse formato de arquivo. Uma outra ferramenta já vista anteriormente, que também pode entender o conteúdo do arquivo, é o *wireshark*.

O resultado completo pode visto no quadro a seguir. Para isso, foi usado o comando “`tcpdump -n -S -t -r hand.pcap`”, em que “`-r hand.pcap`” seleciona o arquivo usado anteriormente para salvar as informações, “`-t`” é usado para não mostrar a coluna de tempo, “`-n`” para não trocar os IP’s por nomes e “`-S`” vamos verificar na descrição dos valores de sequência.

Quadro 5.6 – Sequência da conexão capturada pelo `tcpdump`

```
1. reading from file hand.pcap, link-type EN10MB (Ethernet)
2. IP 127.0.0.1.41580 > 127.0.0.1.30300: Flags [S], seq 4085635198, win
   43690, options [mss 65495,sackOK,TS val 109111951 ecr 0,nop,wscale
   7], length 0
3. IP 127.0.0.1.30300 > 127.0.0.1.41580: Flags [S.], seq 933574057, ack
   4085635199, win 43690, options [mss 65495,sackOK,TS val 109111951
   ecr 109111951,nop,wscale 7], length 0
4. IP 127.0.0.1.41580 > 127.0.0.1.30300: Flags [.], ack 933574058, win
   342, options [nop,nop,TS val 109111951 ecr 109111951], length 0
5. IP 127.0.0.1.41580 > 127.0.0.1.30300: Flags [P.], seq
   4085635199:4085635205, ack 933574058, win 342, options [nop,nop,TS
   val 109113890 ecr 109111951], length 6
6. IP 127.0.0.1.30300 > 127.0.0.1.41580: Flags [.], ack 4085635205, win
   342, options [nop,nop,TS val 109113890 ecr 109113890], length 0
7. IP 127.0.0.1.41580 > 127.0.0.1.30300: Flags [P.], seq
   4085635205:4085635215, ack 933574058, win 342, options [nop,nop,TS
   val 109116384 ecr 109113890], length 10
8. IP 127.0.0.1.30300 > 127.0.0.1.41580: Flags [.], ack 4085635215, win
   342, options [nop,nop,TS val 109116384 ecr 109116384], length 0
9. IP 127.0.0.1.41580 > 127.0.0.1.30300: Flags [F.], seq 4085635215, ack
   933574058, win 342, options [nop,nop,TS val 109116384 ecr
   109116384], length 0
10. IP 127.0.0.1.30300 > 127.0.0.1.41580: Flags [F.], seq 933574058, ack
    4085635216, win 342, options [nop,nop,TS val 109116384 ecr
    109116384], length 0
11. IP 127.0.0.1.41580 > 127.0.0.1.30300: Flags [.], ack 933574059, win
    342, options [nop,nop,TS val 109116384 ecr 109116384], length 0
```

Fonte: Elaborado pelo autor.

Na sequência anterior, estão os passos usados para estabelecer a conexão do TCP e a transmissão das duas informações pelo `ncat` (linhas 5 e 7). Vemos cada passo a seguir.

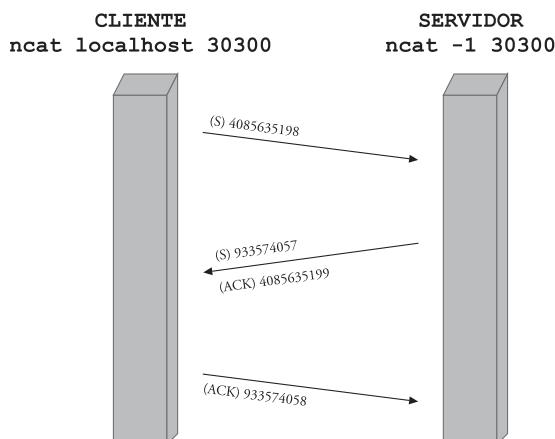
- Na linha 2, o `ncat` cliente solicita que seja estabelecida uma conexão na porta 30300. Para isso, utiliza o bit SYN do cabeçalho do

TCP ligado (**Flags [S]**) e envia um número inicial para identificar a conexão (**seq 4085635198**). Na parte de dados, não é enviado nada (**length 0**).

- b) O servidor responde à solicitação feita na linha 3, indicando que recebeu o pedido do cliente (**ack**) e comprova usando o mesmo número de sequência enviado, acrescentando 1 (**ack 4085635199**). Também envia ao cliente uma sequência inicial da sua comunicação (**seq 933574057**). Assim, temos duas sequências de identificação: uma iniciando com 4085635199, que indica a comunicação do **cliente para o servidor**, e outra iniciando com 933574057, indicando a comunicação do **servidor para o cliente**.
- c) Na linha 4, o cliente responde indicando que também confirma a conexão e comprova que recebeu a sequência do servidor, usando o número recebido e acrescentando 1 (**ack 933574058**).

A figura 5.2 mostra graficamente a sequência inicial da conexão (*HandShake*) comentada acima.

Figura 5.2 – *HandShake* entre *ncat* cliente e servidor



Fonte: Elaborador pelo autor.

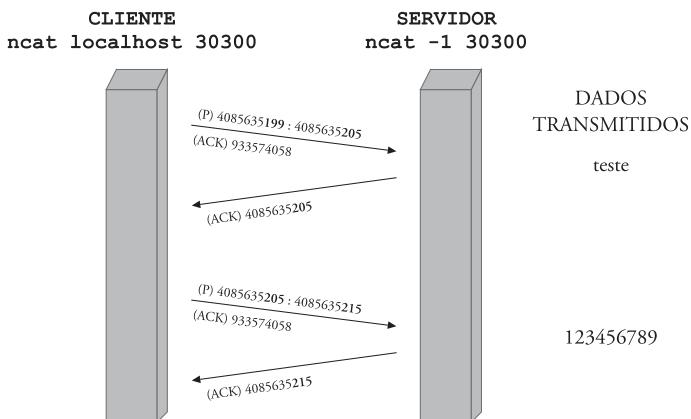
Até a linha 4, foram os passos para estabelecer o início da comunicação do protocolo TCP. Nenhuma informação foi transferida até o momento. Na

linha 6, temos a primeira transmissão do lado cliente: o envio da palavra “*teste*” com 5 letras e mais o código para o <ENTER>. Com isso temos 6 bytes transmitidos (**length 6**), e a numeração da sequência tem o número inicial e final dos bytes apresentados pelo *tcpdump* (sequência inicial  $4085635199 + 6$  bytes do tamanho dos dados =  $4085635205$ ).

Na linha 7, temos a confirmação do recebimento, com o retorno da última sequência recebida (após a soma dos bytes) para o cliente.

Na figura 5.3, temos a sequência de transmissão dos dados feita entre os dois *ncat*.

Figura 5.3 – Transmissão dos dados entre *ncat* cliente e servidor



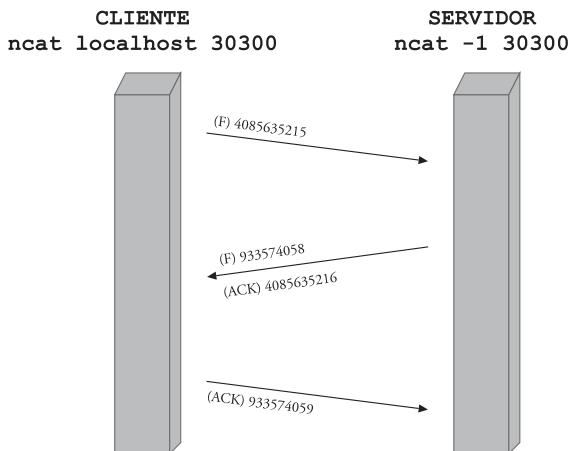
Fonte: Elaborado pelo autor.

Na figura 5.3, é possível verificar o número inicial da transmissão e a confirmação do servidor usando o número final (após a soma dos bytes transmitidos). Também esse último número é usado no início da próxima transmissão do texto “123456789”. É importante lembrar que é enviado junto o indicador do <ENTER> e, por esse motivo, o tamanho é sempre o do texto +1.

Por fim, na linha 9, inicia-se o processo de encerramento da conexão TCP. Ao digitar o CTRL+C na tela do *ncat* cliente, houve o envio de um pacote com o flag FIN habilitado (**Flags [F.]**) com, também, o identificador de confirmação do servidor que foi usado durante todas as transmissões

**(ack 933574058).** O servidor ainda estabelece o processo de encerramento, enviando o seu identificador (**seq 933574058**), e uma confirmação com o número identificador incrementado de 1 (**ack 4085635216**). Finalizando o processo o cliente valida enviando apenas uma confirmação com o número incrementado também em 1 (**ack 933574059**).

Figura 5.4 – Encerramento da conexão



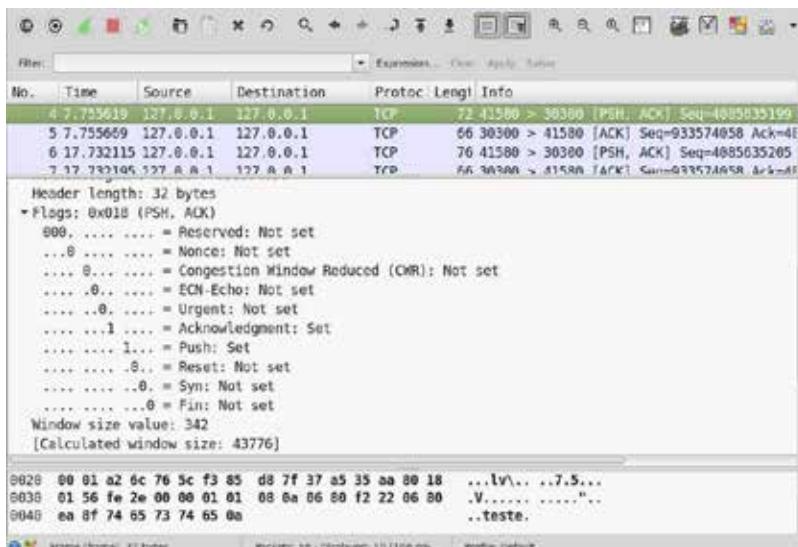
Fonte: Elaborado pelo autor.

A partir do momento em que a transmissão foi salva em um arquivo no formato do *tcpdump*, os mesmos passos podem ser analisados usando a ferramenta *wireshark*, indicando na linha de comando: “*wireshark hand.pcap*”. Por meio dela, pode-se também verificar outras *flags* e opções incluídas no cabeçalho TCP, assim como verificar no painel final os dados transmitidos. Na figura 5.5, temos o mesmo arquivo “*hand.pcap*” com as informações dos *flags* do TCP apresentados e, no painel final, a palavra “**teste**”, que foi transmitida durante a sessão.

O exemplo apresentado, usando o comando *ncat* tanto como cliente quanto como servidor, serve para efeitos didáticos, mas a lógica de captura com o *tcpdump* e as análises feitas passo a passo, com a identificação da situação de cada pacote, as requisições e confirmações feitas pelo TCP, são também vistas nos cenários reais de rede com conexões feitas entre *browsers* e servidores web, ou outros protocolos de aplicação que usarão o TCP como base.

Conhecendo o número da porta usada para o serviço e usando-a nos filtros aplicados para a captura, evita-se a gravação dos outros diversos protocolos e conexões sendo tratadas pela interface. Desta forma concentra-se as informações apenas na conexão que deve ser analisada.

Figura 5.5 – Apresentação do conteúdo de *hand.pcap* no *wireshark*



Fonte: Elaborado pelo autor.

### 5.2.3 Retransmissão

Durante os testes e exemplos do TCP, podem ser verificados os identificadores que são estabelecidos para cada pacote transmitido. Uma continuidade dos números de requisição e confirmação é feita para o protocolo poder perceber caso haja uma interrupção.

Mas ao enviar um pacote com uma sequência numérica, não se pode aguardar um tempo muito longo, pois o pacote enviado pode ter sido perdido, e a confirmação não irá chegar. É preciso indicar um tempo máximo para a espera, chamado de RTO (*Retransmission TimeOut*). Esse valor é inicialmente calculado com base no tempo que os pacotes levam entre a transmissão e o recebimento da confirmação.

Sempre que um pacote é enviado também é iniciado o contador de tempo do TCP. Caso a confirmação seja recebida antes de finalizar o tempo, o contador é zerado e inicia novamente com uma nova transmissão.

## Saiba mais

Os parâmetros utilizados nas conexões podem ser alterados para permitir melhorar o desempenho de um servidor conforme o volume de conexões que ele precisa administrar. Em alguns casos, tempos de espera, espaço de filas de conexões e outras características do código que foi desenvolvido podem melhorar o desempenho. O GNU/Linux pode ser alterado por meio de algumas variáveis do sistema operacional. Por exemplo, para saber quanto tempo o sistema espera para liberar uma conexão que foi finalizada, pode-se usar `"sysctl net.ipv4.tcp_fin_timeout"` e ver o resultado (o padrão é 60 segundos). Caso o servidor tenha muitas conexões, esse valor pode ser diminuído para que sejam liberadas mais rápido.



Caso a confirmação não seja entregue antes de finalizar o contador, o TCP faz a retransmissão do pacote original. Após várias tentativas de retransmissão do pacote original e não recebendo confirmação de nenhuma delas, o aplicativo interrompe a conexão.

Para simular esse processo, utilizou-se novamente o `ncat`, mas, dessa vez, em outro dispositivo (192.168.0.12) e com a porta 30300 em espera. Foi transferida a palavra “teste” para estabelecer a conexão e executar a primeira transferência, usando o mesmo procedimento dos exemplos anteriores.

Quadro 5.7 – Conexão remota com interrupção

LADO CLIENTE	LADO SERVIDOR
<pre># ncat 192.168.0.12 30300 teste outro teste  Ncat: No route to host.</pre>	<pre># ncat -l 192.168.0.12 30300 teste</pre>

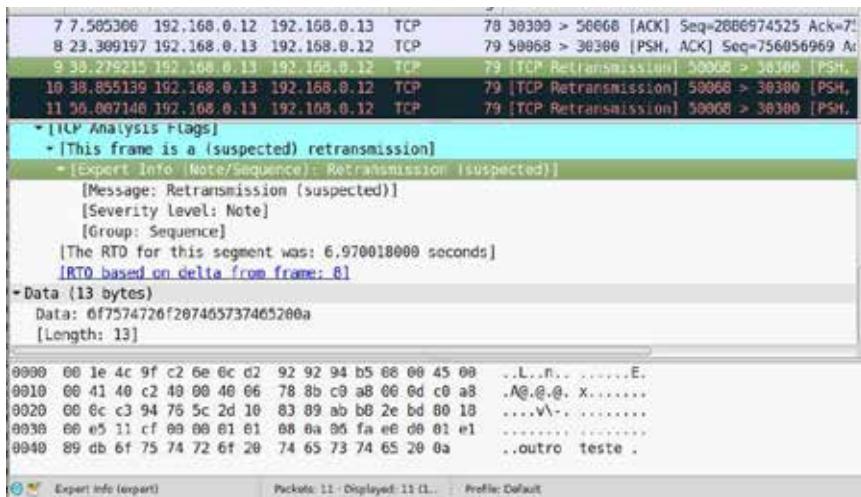
Fonte: Elaborado pelo autor.

Durante o teste, todos os pacotes foram capturados e salvos com o comando “`tcpdump port 30300 -w retransmit.pcap`”.

Antes de teclar o <ENTER> no lado cliente para enviar o segundo texto (“*outro teste*”) para o dispositivo remoto, foi desconectada a interface sem fio do lado servidor. Após desligada a interface e pressionado o <ENTER>, o *ncat* do lado cliente aguarda alguns minutos (em torno de 10) até mostrar a mensagem “**Ncat: No route to host**”.

Na figura a seguir, temos a tela do *wireshark* identificando por cores as linhas com as retransmissões e as informações do cabeçalho da primeira retransmissão, indicando o RTO.

Figura 5.6 – Apresentação do conteúdo de *hand.pcap* no *wireshark*



Fonte: Elaborado pelo autor.

Na figura anterior, temos, no painel superior, a linha 8, em que está a indicação da primeira transmissão do texto “*outro teste*” com o flag PSH indicado no cabeçalho.

Em seguida, nas linhas 9, 10 e 11, estão as retransmissões desse texto e, no painel final, está o texto sendo retransmitido pela linha 9.

## 5.3 UDP (User Datagram Protocol)

O protocolo UDP, também utilizado na camada de transporte, não tem as características e controles usados no TCP. Podemos constatar no cabeçalho mostrado a seguir que a simplificação se estende aos campos definidos para o controle do protocolo.

Basicamente, o protocolo envia e recebe os dados e encaminha ao protocolo da aplicação. Não é feito troca de identificadores, nem controle de sequência, como no TCP. Caso um pacote se perca no caminho ou chegue em ordem trocada, o UDP não trata.

### 5.3.1 Cabeçalho UDP

O cabeçalho do UDP tem um tamanho de apenas 8 bytes e usa os mesmos dois campos iniciais para os endereços de porta de origem e destino, como o TCP.

**Figura 5.7 – Cabeçalho UDP**

16 BIT (porta de origem)	16 BIT (porta de destino)
16 BIT (tamanho UDP)	16 BIT (checagem UDP)
Dados	

Fonte: Elaborado pelo autor.

Em seguida, um campo de 16 bits para identificar o tamanho do cabeçalho é somado ao total de dados do pacote. Depois, há um campo também de 16 bits para armazenar o cálculo de conferência de integridade feito com o cabeçalho, os dados, e inclui os endereços IP's.

Considerando o tamanho pequeno do cabeçalho, a ausência de controles de identificadores, retransmissões, o UDP é processado de maneira mais rápida que o TCP e é usado justamente quando a velocidade tem um peso

maior que a garantia de entrega (aplicações de tempo real). Nesses casos, o protocolo da aplicação faz os controles de que necessita.

## 5.4 Utilitários

Neste capítulo, tivemos o uso do comando *ncat* para simulação dos cenários de conexão via TCP. Temos, na sequência, dois comandos que podem auxiliar no processo de diagnóstico envolvendo a camada de transporte: o comando *Nping* e o comando *Nmap*.

Ambos pacotes podem ser instalados a partir da mesma ferramenta, usando a linha de comando “*apt-get install nmap*”. Tanto o *Ncat*, quanto *Nping* e o *Nmap* são mantidos pelo mesmo grupo de desenvolvedores na Internet.

### 5.4.1 Nping

O utilitário *Nping* tem função semelhante ao *ping* normal, em que é enviado um pacote ao dispositivo de destino e, ao receber uma resposta, consegue-se estabelecer que todo o caminho entre a origem e o destino está alcançável. O problema é que muitas redes acabam filtrando a passagem de pacotes ICMP enviados pelo *ping* normal. Nesses casos, é necessário que o teste de conexão possa ser feito usando outro tipo de protocolo, mas com a mesma flexibilidade. Ao poder indicar o uso de protocolos alternativos ao ICMP, como o TCP e UDP, o teste de conexão pode ser feito mesmo com os filtros ICMP na rede.

No exemplo a seguir, o *Nping* foi utilizado para testar a resposta de um servidor WEB (porta 80), sem a necessidade de utilizar um browser para isso. Foi usado o comando “*nping --tcp -p 80 localhost -c3*” em que “**--tcp**” indica que deve ser usado o protocolo TCP para o teste, “**-p 80**” indica que será testada a porta 80 no dispositivo de destino, “**localhost**” informa o destino (neste caso a interface *loopback*, indicando que o servidor web está na própria máquina) e “**-c3**”, indicando que serão feitas apenas 3 tentativas.

Quadro 5.8 – Resultado apresentado pelo *Nping*

```

1. Starting Nping 0.6.40 ( http://nmap.org/nping ) at 2016-10-12 17:18 BRT
2. SENT (0.0299s) TCP 127.0.0.1:52072 > 127.0.0.1:80 S ttl=64 id=16361
   iplen=40 seq=2902561195 win=1480
3. SENT (1.0314s) TCP 127.0.0.1:52072 > 127.0.0.1:80 S ttl=64 id=16361
   iplen=40 seq=2902561195 win=1480
4. RCVD (1.0317s) TCP 127.0.0.1:80 > 127.0.0.1:52072 SA ttl=64 id=0
   iplen=44 seq=1819106696 win=43690 <mss 65495>
5. SENT (2.0329s) TCP 127.0.0.1:52072 > 127.0.0.1:80 S ttl=64 id=16361
   iplen=40 seq=2902561195 win=1480
6. RCVD (2.0333s) TCP 127.0.0.1:80 > 127.0.0.1:52072 SA ttl=64 id=0
   iplen=44 seq=1834754864 win=43690 <mss 65495>
7. Max rtt: 0.158ms | Min rtt: 0.131ms | Avg rtt: 0.144ms
8. Raw packets sent: 3 (120B) | Rcvd: 2 (88B) | Lost: 1 (33.33%)
9. Nping done: 1 IP address pinged in 2.05 seconds

```

Fonte: Elaborado pelo autor.

No quadro 5.8, verifica-se um exemplo de envio (SENT) do pacote de teste (linhas) e a resposta do servidor (RCVD). No final, assim como o comando *ping* normal, são apresentadas uma estatística resumida do teste das 3 tentativas, com os tempos máximo (Max), mínimo (Min) e médio (Avg) entre o envio e a recepção, e a quantidade enviada e recebida com o percentual de perda (**Lost: 1 (33.33%)**).

No processo de envio do pacote de teste, o *Nping* estabelece a conexão usando o processo de *HandShake* e, logo em seguida, envia um pacote com o flag de reinício da conexão (RESET). Pode-se testemunhar essa situação no trecho capturado no quadro 5.9. Na linha 3, o indicador de reinicio (**Flags [R]**) da conexão. Esse processo de envio e reinicio acontece nas 3 tentativas de conexão parametrizadas no comando *Nping* acima.

Quadro 5.9 – Resultado do *Nping* capturado usando *tcpdump*

```

1. IP 127.0.0.1.52072 > 127.0.0.1.80: Flags [S], seq 2902561195, win 1480,
   length 0
2. IP 127.0.0.1.80 > 127.0.0.1.52072: Flags [S.], seq 1819106696, ack
   2902561196, win 43690, options [mss 65495], length 0
3. IP 127.0.0.1.52072 > 127.0.0.1.80: Flags [R], seq 2902561196, win 0,
   length 0

```

Fonte: Elaborado pelo autor.

Apesar de o exemplo mostrar a conexão por meio do TCP, principal protocolo visto neste capítulo, também podem ser criados pacotes usando UDP, ARP e ICMP.

### 5.4.2 Nmap

A ferramenta *Nmap* é uma das mais utilizadas para fazer varredura em redes remotas. Apesar de possuir muitas funcionalidades voltadas à segurança e à exploração de vulnerabilidades, que não serão abordadas nesta sessão, o *Nmap* pode auxiliar o administrador da rede ao conseguir fazer um levantamento dos dispositivos ativos nela.

O *Nmap* possui um mecanismo para identificar as portas em uso nos dispositivos acessíveis na rede. No quadro 5.10 pode ser verificada a listagem das principais portas em uso no dispositivo remoto em 192.168.0.12. Para incluir portas específicas que o *Nmap* não lista por padrão, pode-se informar por meio de parâmetros na linha de comando com a opção “*-p*”, por exemplo: “*-p 30000-35000*”. Na linha 1, foi usada como parâmetro apenas a eliminação da conversão dos nomes dos dispositivos (“*-n*”).

Quadro 5.10 – Resultado de varredura *Nmap* com identificação das portas

```
1. # nmap 192.168.0.12 -n
2.
3. Starting Nmap 6.40 ( http://nmap.org ) at 2016-10-13 00:39 BRT
4. Nmap scan report for 192.168.0.12
5. Host is up (0.0016s latency).
6. Not shown: 996 closed ports
7. PORT      STATE SERVICE
8. 22/tcp    open  ssh
9. 139/tcp   open  netbios-ssn
10. 445/tcp  open  microsoft-ds
11. 3000/tcp  open  ppp
12. MAC Address: 00:1E:4C:9F:C2:6E (Hon Hai Precision Ind.Co.)
13.
14. Nmap done: 1 IP address (1 host up) scanned in 21.92 seconds
```

Fonte: Elaborado pelo autor.

Como vimos neste capítulo, não é obrigatório que sejam usadas as portas dos dispositivos para os protocolos reservados na base de dados da IANA (2016). Nesses casos, além das portas, é necessário pesquisar qual o serviço efetivo que está associado nas portas do dispositivo remoto. Para isso, é uti-

lizada a opção “*-sV*”. No quadro 5.11, temos o resultado da aplicação dessa opção na varredura do mesmo dispositivo pelo *Nmap*.

Quadro 5.11 – Varredura *Nmap* com identificação dos serviços nas portas.

```

1. mat-cap5 # nmap -n -sV 192.168.0.12
2.
3. Starting Nmap 6.40 ( http://nmap.org ) at 2016-10-13 00:49 BRT
4. Nmap scan report for 192.168.0.12
5. Host is up (0.50s latency).
6. Not shown: 996 closed ports
7. PORT      STATE SERVICE      VERSION
8. 22/tcp     open  ssh          (protocol 2.0)
9. 139/tcp    open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
10. 445/tcp   open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
11. 3000/tcp   open  ntop-ntp  Ntop web interface 4.99.3
12. 1 service unrecognized despite returning data. If you know the
    service/version, please submit the following fingerprint at
    http://www.insecure.org/cgi-bin/servicefp-submit.cgi :
13. SF-Port22-TCP:V=6.40%I=7%D=10/13%Time=57FF0476%P=x86_64-pc-linux-gnu
    tr(NUL
14. SF:L,29,"SSH-2\0-OpenSSH_6\.2p2\x20Ubuntu-6ubuntu0\.4\r\n");
15. MAC Address: 00:1E:4C:C2:6E (Hon Hai Precision Ind.Co.)
16.
17. Service detection performed. Please report any incorrect results at
    http://nmap.org/submit/ .
18. Nmap done: 1 IP address (1 host up) scanned in 32.81 seconds

```

Fonte: Elaborado pelo autor.

Nessa última varredura, podemos analisar o resultado da linha 11 e compará-lo com o resultado da linha 11 do quadro 5.10. Na primeira varredura, apenas foi identificada a porta que estava ativa e foi associada ao serviço padrão que utilizaria essa porta.

Na segunda varredura, foi pesquisado o serviço que efetivamente estava utilizando a porta 3000. Antes, estava associado ao protocolo *ppp* e, ao verificar a real utilização da porta, percebeu-se que está associada ao aplicativo *ntop*, visto em capítulos anteriores de softwares de monitoramento de redes.

## Síntese

Neste capítulo, aprofundamos o conhecimento da camada de transporte, identificando o funcionamento de seus dois protocolos, TCP e UDP. Avaliamos como o TCP consegue garantir que os pacotes sejam recebidos e reorganizados na ordem correta.

Também analisamos cenários criados com o comando *ncat* e inspecionamos com o *tcpdump* cada pacote transmitido para estabelecer a conexão e transferir os dados entre dois dispositivos.

O uso dos utilitários *Nping* e *Nmap* foi detalhado para que eles possam ser acrescentados ao grupo de ferramentas a serem usadas pelo profissional de redes.

## Atividades

1. Como pode ser distinguida uma conexão entre dois dispositivos, sendo que o equipamento de destino apresenta mais de um serviço sendo disponibilizado? Explique quais os parâmetros usados para identificar unicamente a conexão.  
 Para encontrar os serviços de WEB ou SSH no dispositivo remoto, precisamos do DNS, pois essas portas são geradas aleatoriamente.  
 O TCP identifica todo pacote enviado usando um número de sequência que é usado na confirmação de recebimento.  
 O *Ncat* pode simular o serviço de um servidor ao utilizar a opção “-l”.  
 O TCP tem uma sequência de pacotes específica para finalizar uma conexão
3. Qual o impacto no protocolo TCP se não existisse um contador de tempo máximo de espera (*timeout*)?
4. Considerando as funções já existentes no *ping*, qual o motivo de utilizar o comando *Nping* para executar o teste de uma conexão remota?

# 6

## Camada de Aplicação

NESTE CAPÍTULO, ESTUDAREMOS a camada de aplicação da arquitetura TCP/IP. Diferente das camadas anteriores, nas quais o envolvimento com os protocolos durante a conexão era feito pelo sistema operacional, os protocolos dessa camada são utilizados diretamente por aplicativos. Dessa forma, o usuário acaba tendo uma interação maior com os protocolos, uma vez que ele executa aplicativos que vão criar os serviços de comunicação de que necessita.

SÃO DIVERSOS os protocolos de aplicação e novos são desenvolvidos periodicamente. Isso porque são desenvolvidas novas aplicações e muitas delas necessitam de formas diferentes de encapsular e controlar as informações, sejam texto, imagens, som ou arquivos binários. Muitos casos de novas aplicações também usam protocolos já desenvolvidos, mas com novas formatações.

COMO REFERÊNCIA PARA o estudo dos protocolos de aplicação, utilizaremos o NTP (*Network Time Protocol*) e o HTTP (*Hiper-text Transfer Protocol*). Este e o próximo capítulo possibilitarão uma visão abrangente dos protocolos e da arquitetura utilizada como base para o desenvolvimento de sistemas web.

## 6.1 Protocolos de aplicação

A camada de aplicação, como o próprio nome já indica, estabelece uma forma para solucionar um aspecto da comunicação entre um receptor e um emissor. Seja o transporte de um texto científico, o vídeo dos primeiros passos de uma criança ou o resultado de uma pesquisa em um banco de dados, a camada de aplicação, por meio de um software desenvolvido para esse objetivo, busca captar a informação no emissor, ou os comandos que precisam ser enviados, e passa para a próxima camada da pilha TCP/IP, de modo que seja preparada para ser enviada até o outro ponto.

Ao ser recebido no receptor, o protocolo de aplicação consegue reinterpretar a informação e apresentar o seu significado: um texto na tela, um comando para um telescópio, o acionamento de um alarme, um vídeo em tempo real.

Alguns protocolos de aplicação já foram comentados nos capítulos anteriores, pois tratam de funcionalidades executadas por alguns serviços usados no próprio funcionamento da rede: **DNS** (*Domain Name System*), para a conversão de nomes para endereços IP, **DHCP** (*Dynamic Host Configuration Protocol*), para distribuição de endereços IP, **SNMP** (*Simple Network Management Protocol*), para a coleta de informações de gerenciamento remotamente. Outros protocolos também são utilizados em vários cenários da Internet e nas redes locais, como **FTP** (*File Transfer Protocol*), usado na transferência de arquivos, **SMTP** (*Simple Mail Transfer Protocol*), usado para envio de e-mails, e **TLS** (*Transport Layer Security*), usado para criptografar conexões.

Os protocolos da camada de aplicação estão associados ou a algum serviço que executa em 2.º plano no sistema operacional (normalmente chamado de *service*, em sistemas Windows, ou *daemon* em sistemas do tipo Unix), ou são manipulados pelo próprio usuário ao executar um aplicativo no sistema operacional e ao interagir com ele, estabelecendo uma comunicação com um dispositivo ou com outro usuário.

Como muitos outros protocolos vistos até então, existem documentações internacionais padronizadas que permitem que sejam desenvolvidos aplicativos (ou serviços) que utilizam um desses protocolos para a transmissão da informação. Esses são protocolos **abertos**. Temos, também, protocolos que executam um processo de empacotamento e transmissão específico de

uma empresa, podendo ou não divulgar formas de interação com outros serviços e protocolos. São os protocolos **proprietários**.

É nessa camada que a informação propriamente dita estará sendo trabalhada, e são estes protocolos que, aos serem entendidos, permitem a inspeção do conteúdo da transmissão e a preocupação com a privacidade da comunicação. Essa informação tem evoluído do texto puro ou texto formatado dos primeiros aplicativos para voz, imagens e vídeos, que, por sua vez, exigem grande capacidade de transmissão de dados e/ou baixa latência (atraso) de transmissão.

Dentro da rede, as aplicações precisam calcular o tempo entre mensagens e a validade das informações. Um importante auxiliar desse processo de manutenção do tempo é o protocolo NTP (*Network Time Protocol*).

## 6.2 NTP (Network Time Protocol)

Segundo Mills (2016), o NTP é um dos protocolos da camada de aplicação usados desde o início da Internet e é considerado um dos mais antigos protocolos em uso contínuo.

Nele, a informação base é o tempo, que deve ser sincronizado entre os dispositivos, atuando na manutenção da infraestrutura da rede. Muitos serviços são influenciados pela contagem do tempo, pois precisam avaliar se uma informação ainda é válida, ou registrar em que momento foi iniciado ou finalizado um processo.

Quando os dispositivos têm diferença de horário entre si, um dado que chega ao receptor pode ser erroneamente classificado, analisado fora do prazo ou antecipar a execução de algum comando. O rastreamento da comunicação pode ser interpretado de forma inconsistente ao serem coletados os registros (logs) do processamento dos pacotes entre os dispositivos, seja por uma questão de análise de desempenho ou de validação de segurança.

A sincronização dos relógios dos computadores permite a correta interpretação do tempo entre vários pontos da rede. O NTP permite essa sincronia por meio da rede e, para isso, avalia também o atraso do sinal entre o emissor até chegar no receptor, considerando que nesse trajeto pode haver a passagem por diferentes dispositivos de interligação.

O tempo em um dispositivo é registrado internamente por meio de seu sistema operacional e do hardware responsável pela contagem do tempo. Um cristal de quartzo é o método utilizado para esse cálculo. Quando usado em um circuito eletrônico, ele vibra em uma frequência de 32.768 Hz. Considerando que um Hertz (Hz) equivale a um ciclo por segundo, dividindo a frequência por 32.768, temos um segundo.

Mas o tempo não é mantido no formato a que estamos acostumados, com horas, dias, meses e anos. É armazenada a contagem de segundos desde uma data de referência. Sempre que for necessária a visualização por uma pessoa, esse período de tempo é convertido. Para o computador e demais dispositivos, para calcular o tempo, usa-se a diferença dos números coletados nos dois intervalos.

Enquanto o uso do cristal de quartzo atende às necessidades dos dispositivos de uso comum, contadores de maior precisão são utilizados para referência de tempo em países e cenários onde é necessária maior exatidão, como sistemas de alta velocidade, sinais de navegação e posicionamento. Nesses casos, relógios atômicos (Oscilador de Césio) podem ser usados como fonte de referência para o cálculo mais exato do tempo, pois, conforme NIST (2016), o Sistema Internacional de Medidas (SI – International System of Units) padroniza o segundo como sendo o equivalente a 9.192.631.770 ciclos de radiação do Césio 133.

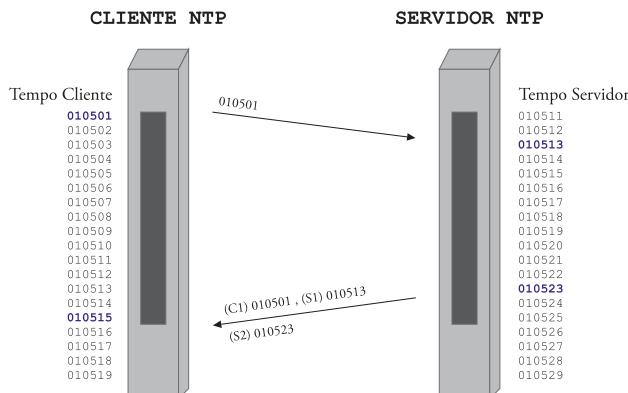
---

### 6.2.1 Sincronia remota

Para realizar o ajuste no tempo, o NTP acessa um serviço de rede que permite o uso de um relógio de maior precisão, calcula a diferença entre o

cliente e esse serviço e desconta o atraso da rede no processo. Mas esse cálculo de ajuste deve ser feito várias vezes para que o conjunto de valores identificados possa ser mais preciso.

Figura 6.1 – Transmissão do tempo entre cliente e servidor



Fonte: Adaptado de NTPBR (2016).

No exemplo da figura 6.1, para identificar a diferença entre o cliente e o servidor, o NTP coleta o *timestamp* (número de segundos registrados pelo computador) do cliente e transmite para o servidor. Na figura 6.1, o tempo em azul, em cada lado, identifica o momento da transmissão. O servidor identifica o momento em que recebeu a informação (“010513” em azul). Em um segundo momento, o servidor envia para o cliente o tempo registrado originalmente (C1 – “010501”), o tempo registrado no recebimento da primeira mensagem (S1 – “010513”) e o tempo no momento da transmissão da segunda mensagem (S2 – “010523”).

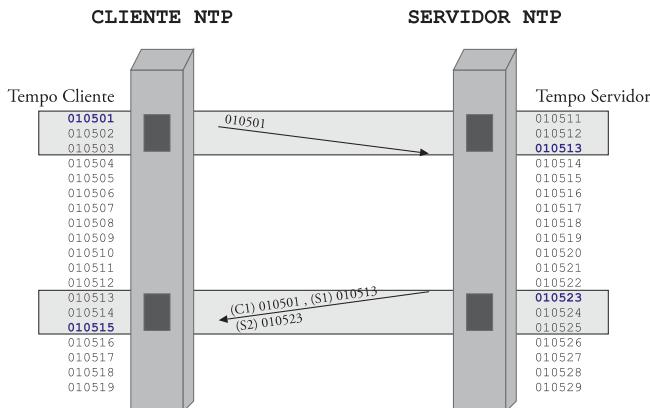
O cliente, ao receber os três registros de tempo, novamente armazena o tempo em que recebeu a informação (C2 – “010515”). Com essas quatro medidas no total, o cliente pode calcular o atraso da transmissão e a diferença entre o cliente e o servidor.

Para calcular o atraso da transmissão, o cliente utiliza a diferença entre o tempo total do envio e o recebimento das duas mensagens do lado servidor com o total do intervalo do seu lado. O resultado é o intervalo total entre a saída do pacote e a chegada das mensagens.

Por exemplo, no lado cliente ( $010501 - 010515 = 14$ ) e, no lado servidor, ( $010513 - 010523 = 10$ ). No final, são quatro intervalos ( $14 - 10 = 4$ ) e é feita a divisão por dois, ficando o atraso em dois intervalos de tempo para cada transmissão.

Na figura 6.2, no fundo cinza, fica visível o tempo que o pacote levou para sair do cliente e chegar ao servidor.

Figura 6.2 – Atraso da transmissão do tempo



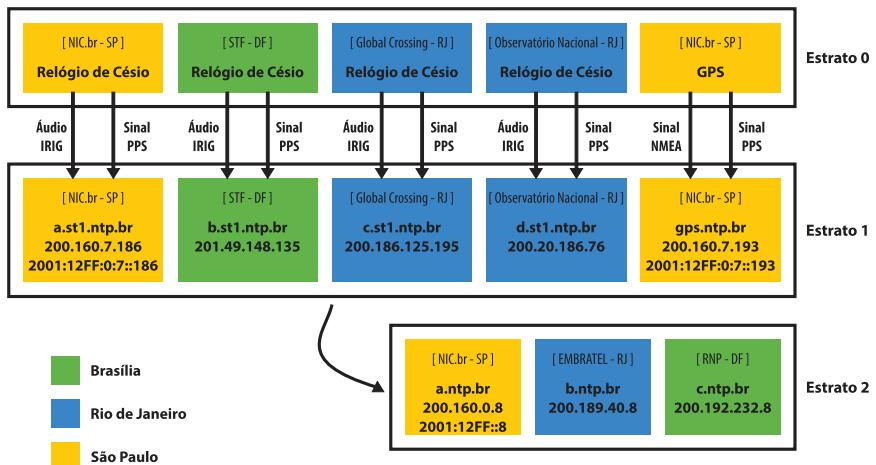
Fonte: Adaptado de NTPBR (2016).

Depois do atraso é feito o cálculo da diferença entre o servidor (que tem o melhor controle de tempo) e o cliente. Continuando o exemplo, a diferença de tempo entre os dois pontos na primeira transmissão entre servidor e cliente é ( $010513 - 010501 = 12$ ) e ( $010523 - 010515 = 8$ ) na segunda transmissão. As duas diferenças são somadas e divididas pelo atraso de cada transmissão ( $(12 + 8) / 2 = 10$ ). O cliente precisa corrigir em dez intervalos o seu horário.

Esse cenário se repete várias vezes e mesmo já tendo calculado um valor de ajuste, a diferença não é aplicada de uma única vez e, sim, de acordo com algoritmos de ajustes.

No Brasil, existem servidores públicos que são mantidos pelo NIC.br (Núcleo de Informação e Coordenação do Ponto BR) e que podem ser configurados nos computadores clientes e, assim, ter o horário ajustado com a Hora Local Brasileira (HLB).

Figura 6.3 – Estrutura dos servidores de tempo no Brasil



Fonte: NTP.br (2016).

Os servidores da figura 6.3 estão interligados de maneira hierárquica, sendo que os do primeiro nível (chamado de estrato 0) estão conectados a osciladores de alta precisão.

Ao configurar um dispositivo, basta indicar um dos servidores públicos “a.ntp.br”, “b.ntp.br” ou “c.ntp.br”, e o sistema que tiver o NTP como protocolo de sincronização de horário vai ajustar o relógio local. Para aumentar a disponibilidade do acesso a esses servidores, pode ser usado o “*pool.ntp.br*”, que aponta para um conjunto aleatório de servidores, garantindo que sempre seja encontrado um servidor ativo.

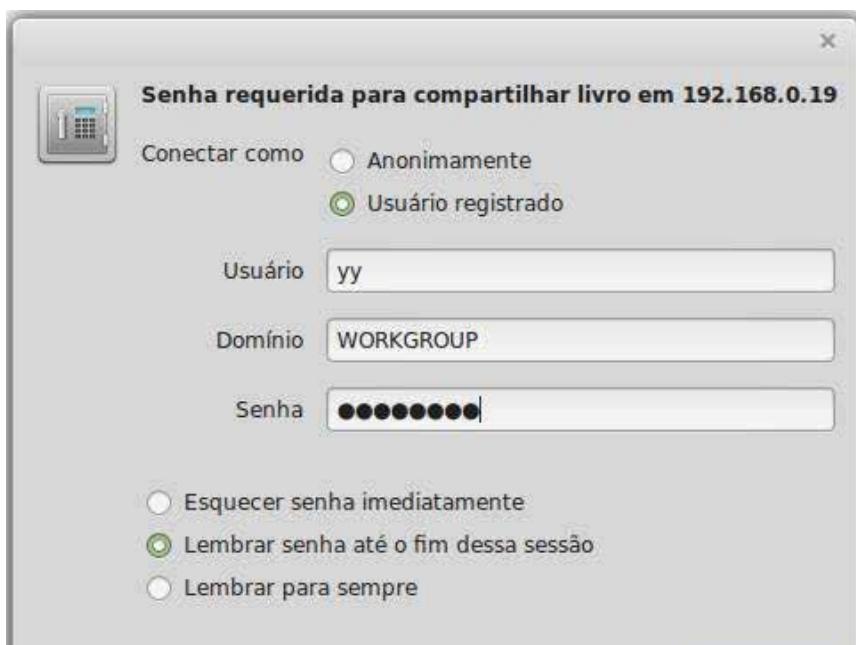
## 6.3 Protocolo HTTP

Um dos protocolos que permitiu a criação do ecossistema de informação mundial em que se tornou a Internet é o protocolo de aplicação HTTP, que tem sido usado para transmitir diversos tipos de informação, mas, para compreendermos algumas tecnologias que estão relacionadas a esse processo de divulgação, armazenamento e processamento de informações nesse grande ambiente virtual, precisamos entender sua proposta inicial e o objetivo que levou ao seu desenvolvimento.

### 6.3.1 Ambiente WEB (WWW ou World Wide Web)

No Centro Europeu de Pesquisa Nuclear (CERN), na década de 90 do século XX, havia diversos projetos que envolviam equipes com nacionalidades diferentes e uma grande quantidade de documentação. O acesso a esses documentos e às informações sobre o andamento das pesquisas era complexo, pois envolviam formatos, sistemas e protocolos diferentes. Tim Berners-Lee propôs, em 1989, uma forma de estruturar o acesso a esses documentos por meio de Hipertexto (um conceito que estava começando a ser disseminado), mantendo a documentação em seu formato original, mas podendo ser acessada por meio de ligações (links) em uma página de texto. Durante o projeto, foram desenvolvidas as tecnologias base da Internet atual, como o protocolo HTTP, servidores HTTP, linguagem de marcação HTML e Navegadores (Browsers), que ainda serão estudados neste capítulo.

Figura 6.4 – Primeiro browser



Fonte: Browser (2016).

Na figura 6.4, é possível identificar o processo de interligar os documentos usando o primeiro software (na época chamado de “*worldwideweb*”) desenvolvido no CERN, que permitia a navegação e a edição das páginas.

Para fazer o acesso entre as páginas dos documentos, foi criado um mecanismo de endereçamento, chamado de URL (*Uniform Resource Locators*, ou localizador de recursos padronizados).

### 6.3.2 URL (Uniform Resource Locators) / URI (Universal Resource Identifier)

A URL é uma forma de localizar um objeto usando um protocolo existente em qualquer local na Internet. É uma forma específica da URI (*Universal Resource Identifier*, ou identificador universal de recursos), que, por sua vez, é um formato genérico de identificação de um objeto. A URI não está limitada a um endereço de rede.

Durante este capítulo, será utilizado o URL como forma de identificar os objetos, sua localização e as interligações. Um exemplo de URL pode ser: “<http://www.ietf.org/rfc/rfc3986.txt>” ou “<file:///etc/ntp.conf>”.

O formato geral da URL pode ser visto no quadro a seguir e, como vimos nos exemplos anteriores, nem todos os campos são necessários ao indicar uma localização.

Quadro 6.1 – Formato de uma URL

<pre><b>&lt;protocolo&gt;</b> : // <b>&lt;usuário&gt;</b> : <b>&lt;senha&gt;</b> @ <b>&lt;dispositivo&gt;</b> : <b>&lt;porta&gt;</b> / <b>&lt;caminho&gt;</b> / <b>&lt;arquivo&gt;</b> ? <b>&lt;pesquisa&gt;</b></pre>	
<b>&lt;protocolo&gt;</b>	Nome do protocolo a ser utilizado para acesso ao objeto. Exemplos: HTTP, FTP, LDAP, File.
<b>&lt;usuário&gt;</b>	Nome que será usado no mecanismo de autenticação do protocolo, para o acesso ao objeto.
<b>&lt;senha&gt;</b>	A senha que será utilizada para a autenticação do acesso. Considerando que a senha poderá passar aberta pela Internet, não é recomendado seu envio pela URL.

<pre>&lt;protocolo&gt; : // &lt;usuário&gt; : &lt;senha&gt; @ &lt;dispositivo&gt; : &lt;porta&gt; / &lt;caminho&gt; / &lt;arquivo&gt; ? &lt;pesquisa&gt;</pre>	
<b>&lt;dispositivo&gt;</b>	É a localização do serviço a ser acessado. Pode ser um nome, um domínio, um IPv4 ou IPv6.
<b>&lt;porta&gt;</b>	Um número de porta TCP ou UDP. No caso de o protocolo ser HTTP, o número da porta padrão (80) pode ser omitido.
<b>&lt;caminho&gt;</b>	Forma de hierarquizar o acesso à informação. Normalmente associado a uma estrutura de arquivos de um sistema operacional.
<b>&lt;arquivo&gt;</b>	Objeto a ser acessado. Pode ter diferentes formatos, como PDF, Planilha, HTML, MP3.
<b>&lt;pesquisa&gt;</b>	Forma de envio de dados para o servidor. No modo mais comum é feito o envio de um par com o formato de “CHAVE=VALOR”.

Fonte: Elaborado pelo autor com base em RFC1738 e RFC3986(2016).

A utilização da URL permite que uma informação remota seja localizada e associada a uma palavra em um documento, criando a base do Hipertexto, que é um conceito que propõe associar o conhecimento de forma não hierárquica, permitindo uma rápida associação das ideias.

### 6.3.3 Comandos HTTP

O protocolo HTTP foi concebido como um mecanismo simples de transferência de documentos. Possui comandos simples para a solicitação do documento e, enquanto os arquivos Hipertexto foram o objeto central da arquitetura da Web, atualmente podem ser feitas transferências de informação via HTTP que não precisam estar vinculados a um arquivo de Hipertexto.

Os comandos são linhas de texto que são enviadas ao servidor, que, por sua vez, processa o comando e retorna a resposta. Este é o funcionamento básico do protocolo: uma solicitação e uma resposta. Na requisição, são enviados um comando, a localização, a versão do protocolo e uma sequência

de parâmetros. No quadro 6.2, temos o uso do comando *ncat*, visto em capítulos anteriores, sendo usado para o envio do comando “GET” ao servidor e recebendo a resposta.

Quadro 6.2 – Simulação de requisição HTTP usando *ncat*

Solicitação do cliente	Resposta do servidor
# <del>ncat</del> 192.168.0.13 80 GET /teste.html HTTP/1.1	<ol style="list-style-type: none"> <li>1. <del>HTTP/1.1 200 OK</del></li> <li>2. <del>Date: Sat, 15 Oct 2016 18:00:28 GMT</del></li> <li>3. <del>Server: Apache/2.4.7 (Ubuntu)</del></li> <li>4. <del>Last-Modified: Sat, 22 Oct 2016 17:58:23 GMT</del></li> <li>5. <del>ETag: "61-53f77e7a1f8a7"</del></li> <li>6. <del>Accept-Ranges: bytes</del></li> <li>7. <del>Content-Length: 97</del></li> <li>8. <del>Vary: Accept-Encoding</del></li> <li>9. <del>Connection: close</del></li> <li>10. <del>Content-Type: text/html</del></li> <li>11.</li> <li>12. &lt;html&gt;</li> <li>13. &lt;body&gt;</li> <li>14. &lt;p style="color:red"&gt; documento HTML &lt;/p&gt;</li> <li>15. &lt;p&gt; Retorno do GET &lt;/p&gt;</li> <li>16. &lt;/body&gt;</li> <li>17. &lt;/html&gt;</li> <li>18. ^C</li> </ol>

Fonte: Elaborado pelo autor.

Na coluna da direita, a resposta do servidor inclui alguns parâmetros de identificação do HTTP (linhas 1 a 10) e, ao final (a partir da linha 12), o conteúdo do arquivo “teste.html” que havia sido solicitado.

No quadro a seguir, temos a sequência dos comandos que podem ser utilizados nas requisições HTTP.

Quadro 6.3 – Comandos de requisição HTTP

Comando	Descrição
OPTIONS	Identifica quais recursos o servidor tem implementado. Um exemplo de uma resposta a uma solicitação é: “Allow: GET,HEAD,POST,OPTIONS” do servidor httpd Apache.
GET	Principal comando do HTTP. Faz a solicitação a um recurso do servidor.

Comando	Descrição
POST	Envia parâmetros para o servidor e obtém um resultado da execução. Utilizado para envio de informações de formulários e publicação de mensagens. As funções executadas por um POST dependem da implementação do servidor HTTP.
PUT	Envia informação que será armazenada no servidor.
DELETE	Solicita a remoção de um recurso no servidor.
HEAD	Na solicitação feita via comando HEAD, são retornadas apenas as informações de cabeçalho do recurso, mas sem o envio do conteúdo solicitado.
TRACE	Permite receber de volta a solicitação feita ao servidor. Usado para testes. Esse método pode não estar liberado pelo servidor. Nesse caso, uma mensagem do tipo: " <b>&lt;h1&gt;Method Not Allowed&lt;/h1&gt;</b> <b>&lt;p&gt;The requested method TRACE is not allowed for the URL /.&lt;/p&gt;</b> " é recebida como resposta, indicando que o TRACE não é permitido.
CONNECT	Usado para conexão com um serviço de proxy.

Fonte: Elaborado pelo autor com base em RFC2616 (2016) e Tanenbaum; Wetherall (2011).

Os dois principais comandos são o GET e POST, que efetivamente fazem a requisição. O funcionamento deles pode ser simulado por meio de uma conexão texto com um servidor HTTP, como verificado no quadro 6.2.

O comando HEAD, visto no quadro 6.2, e pode ser comparado o retorno com o quadro 6.4.

Quadro 6.4 – Simulação de comando HEAD usando *ncat*

Solicitação do cliente	Resposta do servidor
# <b>ncat 192.168.0.13 80</b> <b>HEAD /teste.html HTTP/1.1</b>	1. <i>HTTP/1.1 200 OK</i> 2. <i>Date: Sat, 15 Oct 2016 18:00:28 GMT</i> 3. <i>Server: Apache/2.4.7 (Ubuntu)</i> 4. <i>Last-Modified: Sat, 22 Oct 2016 17:58:23 GMT</i> 5. <i>ETag: "61-53f77e7a1f1fa7"</i> 6. <i>Accept-Ranges: bytes</i> 7. <i>Content-Length: 97</i> 8. <i>Vary: Accept-Encoding</i> 9. <i>Connection: close</i> 10. <i>Content-Type: text/html</i>

Fonte: Elaborado pelo autor.

Nessa simulação do uso do comando HEAD, verificamos que as mesmas informações **sobre** o conteúdo HTML são retornadas, mas o conteúdo em si não é transferido pelo HTTP.

### 6.3.4 Páginas estáticas

A utilização do Hipertexto para o acesso à documentação científica e de diferentes tipos de conteúdo, depois da liberação para uso comercial das tecnologias, permitiu que informações de diversos tipos fossem disponibilizadas na web.

Esses documentos eram acessados e visualizados nos navegadores (browsers), e a cada interligação que era acessada o conteúdo era trocado e, assim, feitas as navegações pela web.

A formatação desses conteúdos ficou direcionada a uma linguagem de marcação que podia ser utilizada em qualquer editor de texto simples.

Como o conteúdo era atualizado manualmente e não por um sistema computacional, esses documentos eram conhecidos como páginas estáticas.

#### 6.3.4.1 HTML/HTML5

O padrão para a formatação das páginas web foi o HTML (*HyperText Markup Language*, ou Linguagem de Marcação de Hipertexto). Desenvolvido nos primeiros momentos da criação da web no CERN, esse padrão está associado a uma linguagem de marcação de conteúdo que sinaliza qual parte do texto é um conteúdo e qual parte do texto está associada a uma ligação com outros documentos. Também sinaliza o agrupamento do texto em parágrafos e listas.

Com a evolução da capacidade das redes e dos computadores, foram acrescentados mais tipos de conteúdo, como imagens, vídeos e som nas páginas HTML, e a linguagem foi se transformando, mas mantendo a simplicidade na sua utilização.

Para formatar um documento HTML, é necessário utilizar marcadores (TAG) que são comandos intercalados pelos sinais “<” e “>”. Por exemplo, temos <html>, <table>, <i> que são, respectivamente, indicador do documento HTML, formatação de uma tabela e sinal de formatação da fonte de texto em itálico.

Quadro 6.5 – Estrutura de um documento HTML

COMANDOS HTML	RESULTADO NO BROWSER
<p>1. &lt;html&gt;</p> <p>2. &lt;head&gt; &lt;/head&gt;</p> <p>3. &lt;body&gt;</p> <p>4. &lt;h1&gt; Texto em HTML &lt;/h1&gt;</p> <p>5. &lt;p&gt;</p> <p>6. Quantos &lt;i&gt;&lt;b&gt; Switchs &lt;/b&gt;&lt;/i&gt; temos</p> <p>7. na rede interna que participam &lt;br&gt;</p> <p>8. do acesso do sistema administrativo ao</p> <p>9. banco de dados? &lt;br&gt;&lt;br&gt;</p> <p>10. Para entendimento da rede, utilize</p> <p>11. o documento:</p> <p><b>12. &lt;a href="projeto.html"&gt;Projeto da rede&lt;/a&gt;</b></p> <p>13. &lt;/p&gt;</p> <p>14. &lt;/body&gt;</p> <p>15. &lt;/html&gt;</p>	<p><b>Texto em HTML</b></p> <p>Quantos <b>Switchs</b> temos na rede interna que participam do acesso do sistema administrativo ao banco de dados?</p> <p>Para entendimento da rede, utilize o documento: <a href="#">Projeto da rede</a></p>

Fonte: Elaborado pelo autor.

Nesse quadro, temos o texto com as TAGs HTML na esquerda e, na direita, como o navegador o apresenta após a formatação. As TAGs têm a marcação do início e do final do conteúdo. Para isso, o nome da TAG é repetido e acrescentado o sinal de /. No exemplo anterior, na linha 4, a marcação < h1 > indica um tipo de título, e a indicação do final do texto que participa do título é feita pela marcação < /h1 >.

Na linha 6, verificamos que duas TAGs <i> e <b> estão intercaladas. As TAGs <b> e </b>, que indicam **negrito**, estão dentro das TAGs <i> e </i>, que indicam *itálico*. Isso significa que a palavra “switchs” estará formatada com negrito e itálico.

Na linha 12, temos um hiperlink (uma ligação de Hipertexto com outro documento). Ele é indicado pela TAG `<a href="projeto.html">`, que tem um atributo “`href`”, que informa qual documento deve ser acessado. Depois, é utilizado um texto que terá a marcação do hiperlink e a parte final da TAG `</a>`. No lado direito do quadro, vemos o resultado da marcação do hiperlink: o texto “Projeto de rede” está sublinhado, indicando que, quando ele é clicado, será carregado o documento “`projeto.html`”.

Na última versão (HTML5), foram acrescentadas formas de incluir mídias de vídeo e som, gráficos vetoriais e manipulação de pixels por linguagem de scripts.

O uso e a alteração de pixels na tela do navegador tornaram possíveis a implementação direta em HTML da construção de cenários gráficos avançados e objetos 3D.

#### 6.3.4.2 Formulários

Para permitir a interação do usuário e, assim, enviar diferentes combinações de parâmetros para o servidor, temos um conjunto de TAGs que estão associadas à criação de um formulário HTML. Esse formulário coleta essas opções informadas pelo usuário e transfere para o servidor usando um dos métodos que serão estudados mais adiante, permitindo que o sistema do lado servidor possa processar uma resposta condizente com a opção escolhida pelo usuário. No quadro 6.6, um exemplo de formulário está na coluna da esquerda e, na direita, o resultado que o navegador apresenta ao usuário. No formulário, apenas quando é feito o envio (no caso a seguir, com o clique no botão “**Envia**”) é que o servidor recebe as informações.

Quadro 6.6 – Estrutura de um formulário HTML

COMANDOS HTML	RESULTADO NO BROWSER
<pre> 1. &lt;html&gt; &lt;head&gt;&lt;title&gt;&lt;/title&gt;&lt;/head&gt; 2. &lt;body&gt; 3. &lt;form action="pedido.php"&gt; 4. 5. Digite o numero da mesa: 6. &lt;input type="text" name="numero"&gt; 7. 8. &lt;fieldset&gt; 9.   &lt;legend&gt;Indique a forma de pagamento:&lt;/legend&gt; 10.  &lt;input type="radio" name="pag" value="dinheiro" /&gt; Dinheiro &lt;br /&gt; 11.  &lt;input type="radio" name="pag" value="cheque" /&gt; Cheque 12. &lt;/fieldset&gt; 13. Escolha o produto: 14. 15. &lt;select name="ipoprod[]"&gt; 16.   &lt;option value="dog"&gt;Cachorro quente 17.   &lt;option&gt; 18.   &lt;option value="mis"&gt;Misto quente 20.   &lt;option&gt; 21.   &lt;option value="ham"&gt;<b>Hamburger</b> 22.   &lt;/option&gt; 23.   &lt;option value="slg"&gt;Salgado 24.   &lt;option&gt; 25. &lt;/select&gt; 26. 27. &lt;input type="submit" value="Envia"&gt; 28. &lt;/form&gt; 29. &lt;/body&gt; &lt;/html&gt;</pre>	<p>Digite o numero da mesa: _____</p> <p>Indique a forma de pagamento:</p> <p><input type="radio"/> Dinheiro <input type="radio"/> Cheque</p> <p>Escolha o produto:</p> <p>Cachorro quente Misto quente <b>Hamburger</b> Salgado</p> <p>Envia</p>

Fonte: Elaborado pelo autor.

No quadro anterior, na linha 3, a TAG `<form>` é utilizada para indicar o início do formulário. A TAG `<input>` e seus atributos na linha 6 permitem que o usuário possa digitar o número da mesa. E, na linha 21, uma das opções do menu de seleção.

Na linha 27, temos a TAG que cria o botão de envio das informações do formulário e, na linha 28, no final do formulário, a TAG `</form>`.

Para acompanhar o acesso ao servidor e o envio das informações de um formulário, podemos usar um comando que permite uma interação direta com o protocolo HTTP. O comando “curl” envia as solicitações de acordo com o formato do protocolo indicado. Pode usar diversos protocolos, por

exemplo: FTP, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, POP3, POP3S, RTMP, RTSP, SCP, SFTP, SMTP, SMTPS, TELNET e TFTP, mas para cada protocolo devem ser informados os parâmetros correspondentes.

Quadro 6.7 – Acesso ao formulário web com o envio de parâmetros via “curl”

Solicitação do cliente	Resposta do servidor
# curl -d "numero=100&tipo=Compra" http://www.yy.com.br/aulaphp/livro.php	1. <html> 2. Data:22/10 3. 4.  Foram enviados as <u>informações</u> :  5.  Número do pedido: 6. <p style="color:green"> 100 </p> 7.  Tipo Pedido: <b>Compra</b>  8. </html>
# curl -d "numero=15000&tipo=Compra" http://www.yy.com.br/aulaphp/livro.php	1. <html> 2. Data:22/10 3. 4.  Foram enviados as <u>informações</u> :  5.  Número do pedido: 6. <p style="color:green"> 15000 </p> 7. <h1> Número do pedido invalido (maior 10000)</h1> 8.  Tipo Pedido: <b>Compra</b>  9. </html>

Fonte: Elaborado pelo autor.

O comando “curl” permite, além do acesso ao serviço HTTP, enviar informações para o servidor, simulando o preenchimento de um formulário sem a necessidade de abrir a página em um navegador. Esse tipo de acesso pode ser utilizado para testes repetitivos, preenchimento automático de formulários ou análise da transmissão das informações.

Nos exemplos do quadro 6.7, podem ser identificadas duas solicitações ao servidor HTTP, uma em cada linha, as quais simulam o preenchimento do formulário contendo duas variáveis: número e tipo. O comando “curl”, na primeira coluna, envia as variáveis que seriam utilizadas no preenchimento de um formulário web, usando o parâmetro “-d ‘**numero=100&tipo=Compra**’” que informa o servidor que foi preenchido o campo “número” com o valor “100” e o campo “tipo”, com a opção “Compra”. No primeiro exemplo, na coluna de resposta do servidor, está o HTML processado como resposta aos parâmetros enviados. No caso, apenas confirma as informações recebidas. No segundo exemplo, na coluna de resposta, um dos parâmetros (“número”) está

com um valor de “15000”, é processado e entra na condição (“**<b1> Numero do pedido invalido (maior 10000)</b1>**”). Esse tipo de verificação está relacionado ao uso de uma linguagem de programação no lado servidor, que usa o conceito de “páginas dinâmicas”.

### 6.3.5 Páginas dinâmicas

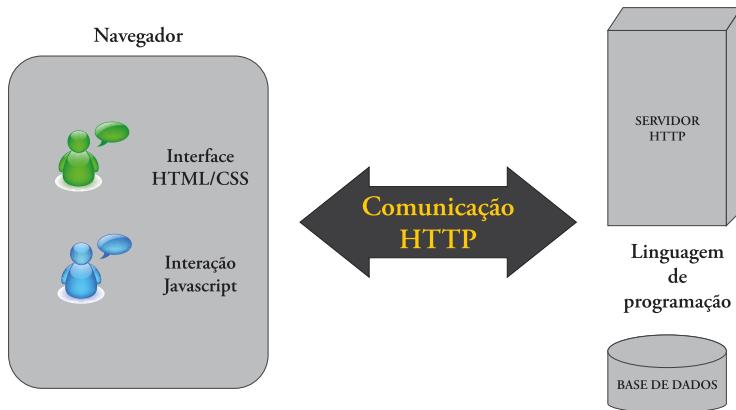
Enquanto as informações constantes em uma página HTML são criadas e necessitam de pouca manutenção, as páginas estáticas cumprem o seu objetivo. Com o uso de formulários e do HTML como interface para sistemas de informação que utilizam bases de dados como repositórios, houve a necessidade de atualizar os conteúdos das páginas de maneira automatizada. Com essa possibilidade, os sistemas poderiam formatar os dados usando o HTML e usar o Hipertexto para a navegação das informações dos bancos de dados.

Nessa situação, o HTML, que é apenas uma linguagem de marcação, e o servidor HTTP, que executa a transferência das informações, não seriam suficientes. Os dados deveriam ser processados e, depois, formatados. Para isso, seria obrigatória a intervenção de uma linguagem de programação que pudesse fazer esse processamento. Para possibilitar o uso de uma linguagem no meio do processo de transferência do HTTP, foi criado o CGI (interface de ligação padrão).

#### 6.3.5.1 CGI (Common Gateway Interface)

O CGI permite interromper o processo do servidor, passar para o sistema operacional o fluxo do processamento e a execução de um programa de computador independente do servidor HTTP. Esse programa executa tarefas e processa as informações (que podem vir de outros programas ou servidores), converte em HTML o resultado e transfere de volta a execução para o servidor HTTP. Este, por sua vez, envia o resultado HTML por meio do protocolo HTTP para o cliente, como pode ser visto na figura 6.5.

Figura 6.5 – Diagrama de comunicação com servidor HTTP



Fonte: Elaborado pelo autor.

No diagrama, é identificada a parte do cliente (navegador) no lado esquerdo e a parte do servidor, no lado direito. Na esquerda, o resultado da solicitação é apresentado pelo navegador, e as tecnologias utilizadas são processadas pelo computador do usuário. Na parte direita, o servidor HTTP concentra as solicitações e, se existir a necessidade de processamento adicional, ele repassa a solicitação para o programa (representado pela linguagem de programação na figura) e este, conforme o algoritmo e os recursos disponíveis (banco de dados, no exemplo), processa a informação e devolve o resultado do processamento para o servidor.

Recebendo o resultado do processamento, o servidor HTTP responde à solicitação do usuário.

Para aumentar o desempenho, alguns servidores HTTP incorporaram a linguagem de programação dentro de seu próprio código. Nessas situações, ocorre a verificação da necessidade do processamento da solicitação e é pas-

sado para a linguagem específica dentro do ambiente do servidor, evitando os mecanismos de troca de processos do sistema operacional.

No próximo capítulo, será detalhado esse processo de automação das páginas dinâmicas.

### 6.3.6 MIME (Multipurpose Internet Mail Extensions)

Toda a comunicação e as informações sendo transferidas via HTTP envolviam um HTML para fazer a interligação com outros tipos de documentos. Mas enquanto tipos de documentos, como HTML, TXT, PDF, RTF, podem ser identificados, a proliferação de mídias e formatos cria uma dificuldade na interpretação do tipo de conteúdo sendo interligado no HTML.

Um mecanismo foi criado para auxiliar o envio de e-mail com conteúdos diferentes do ASCII e possibilitou o envio de caracteres acentuados (necessário em alguns idiomas, como a língua portuguesa do Brasil), fotos em fundo de mensagens e arquivos anexados. Esse mecanismo consistia em acrescentar parâmetros de descrição do conteúdo e na conversão dos bytes não reconhecidos na tabela ASCII.

Muitos dos parâmetros desse padrão foram adotados dentro do HTTP para o uso com conteúdo multimídia.

Primeiramente, é indicada a versão (“*MIME-Version:*”) e, na sequência, alguns parâmetros indicando a codificação usada na transferência, sendo a descrição do conteúdo (“*Content-Type:*”) a informação utilizada para identificação dos diferentes formatos de conteúdo, em um formato de tipo/subtipo.

Em geral, o tipo indicado representa a forma geral do conteúdo (por exemplo: imagem), e o subtipo indica qual o formato específico da imagem (por exemplo: gif). Os tipos básicos foram definidos como: “application”, “audio”, “image”, “message”, “multipart”, “text”, “video”.

No quadro 6.8, podemos verificar alguns tipos de conteúdos e sua descrição.

Quadro 6.8 – Tipos de conteúdos e sua descrição MIME

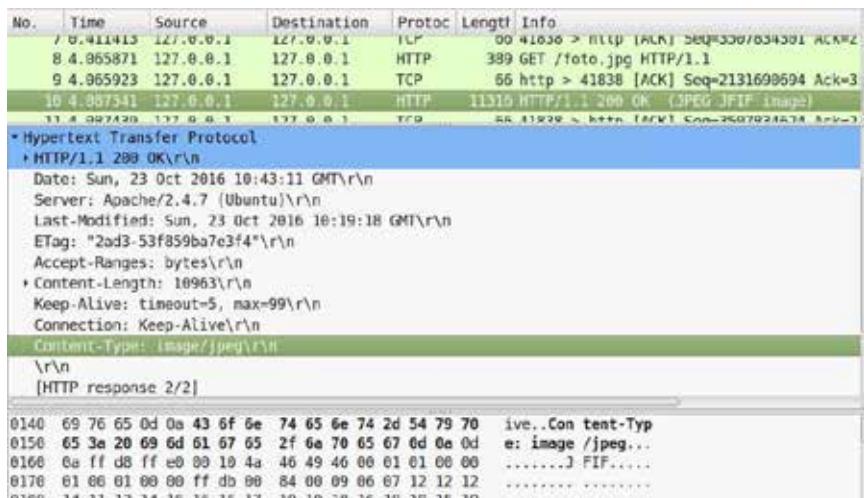
Descrição	Tipos/Subtipos MIME
Texto puro sem formatação.	text/plain

Tipo/Subtipo MIME	Descrição
text/html	Conteúdo com formatação HTML.
application/vnd.ms-excel	Conteúdo para uma aplicação do subtipo planilha Excel.
video/mp4	Conteúdo de vídeo do formato MPEG4.
audio/mpeg	Conteúdo de áudio no formato MPEG3 (MP3)
application/vnd.openxmlformats-officedocument.wordprocessingml.document	Conteúdo para uma aplicação do subtipo processador de texto Word (docx).

Fonte: Elaborado pelo autor.

A quantidade de tipos padrão pode ser estendida conforme registro na IANA (RFC2045, 2016).

Figura 6.6 – Transmissão HTTP com tipo de conteúdo JPEG visualizada com *WireShark*



Fonte: Elaborado pelo autor.

Na figura 6.6, está destacada a informação de conteúdo (“Content-Type: image/Jpeg”) que faz parte da solicitação da linha 10 (ver linha do painel superior do *WireShark*) como indicado pelo retorno do servidor (“HTTP/1.1 200 OK”).

Esse retorno da imagem foi solicitado pelo navegador anteriormente com o comando “GET /foto.jpg HTTP/1.1”), como pode ser verificado na linha 8 do painel superior.

### 6.3.7 Códigos de retorno

Para cada resposta, o servidor indica qual a situação da conexão. Se a requisição foi atendida com sucesso, ele envia essa situação por meio de códigos de retorno. No quadro 6.2, ao retornar a página teste.html solicitada pelo *ncat*, o servidor na primeira linha (“HTTP/1.1 200 OK”) informa o código 200 na resposta. Os códigos estão agrupados em cinco. Os valores iniciando com 1 são informativos, os que iniciam com 2 indicam que a solicitação foi recebida, entendida e aceita, com 3 indicam que mais ações precisam ser feitas pelo software cliente para encontrar o recurso que foi redirecionado, com 4 indicam um erro na solicitação, e os códigos iniciando com 5 indicam um erro no servidor.

### 6.3.8 SPDY/HTTP2

Com o sucesso do protocolo HTTP na Internet, muitas informações são transferidas utilizando esse mecanismo. O volume tem crescido de maneira constante e, com isso, algumas características do protocolo que afetam o desempenho das transmissões ficaram em evidência. O uso de comandos e parâmetros no formato texto e a necessidade de várias conexões a serem feitas para finalizar a transmissão de um conjunto de informações afetam a Internet como um todo. Para atingir essas dificuldades, foi desenvolvida uma nova versão do protocolo: HTTP2 ou HTTP2.0.

Assim como o HTTP1.1, a nova versão utiliza o TCP como meio de transporte. É compatível com os mesmos formatos de URL que estão em

uso e implementa um mecanismo de *upgrade* da conexão ao identificar que o servidor suporta o novo protocolo.

As principais inovações do novo protocolo são a compressão das informações do cabeçalho, a criação de um fluxo de dados entre cliente e servidor (utiliza-se um identificador de 31 bits para a conexão) e a criação de pacotes padronizados (“*frames*”) para cada tipo de informação (pacotes para parâmetros – HEADERS, dados – DATA, priorização – PRIORITY, finalização de fluxo – RST\_STREAM, configuração – SETTINGS, envio de servidor – PUSH\_PROMISE, teste – PING, cancelamento – GOAWAY, fluxo – WINDOW\_UPDATE e continuidade – CONTINUATION).

Com a disseminação desse novo formato de conexão do protocolo HTTP2, a expectativa é de maior desempenho na transmissão de grandes volumes de dados.

### 6.3.9 DASH (Dynamic Streaming over HTTP)

Em algumas situações, o uso do protocolo HTTP não está ligado diretamente à transmissão de informações por meio de Hipertexto. Esse protocolo acaba sendo usado como base de outro formato de transmissão dos dados, como acontece, por exemplo, com o DASH que utiliza o HTTP para transmitir vídeos para uma das maiores empresas de serviço de vídeo na web (ver, no primeiro capítulo, o volume de dados da Netflix na Internet).

Esse mecanismo utiliza um documento XML (linguagem de marcação semelhante ao HTML) para definir os trechos da mídia a serem transmitidos por meio de requisições HTTP. Esse arquivo é chamado de MPD (*Media Presentation Description*, ou descrição de apresentação de mídia) e contém informações sobre a mídia que será transmitida, como resolução do vídeo, tempo e legendas.

Esse padrão foi desenvolvido com a participação de várias grandes empresas, como Microsoft, Apple, Netflix, Qualcomm, Ericsson, e pelo fato de poder usar a mesma infraestrutura e servidores utilizados pelo HTTP consegue ser flexível na sua implementação.

## 6.4 Utilitários

As ferramentas utilizadas neste capítulo para as demonstrações práticas foram apresentadas em capítulos anteriores, assim como os procedimentos de instalação baseados no GNU/Linux da distribuição Mint. A exceção é o utilitário “curl”, que foi utilizado para efetivar o acesso via HTTP ao servidor e que pode ser instalado usando o comando “*apt-get install curl*”. O comando “*wget*” já vem pré-instalado em algumas distribuições GNU/Linux, mas pode ser instalado com o comando “*apt-get install wget*” caso não esteja.

### 6.4.1 WGET

O utilitário “*wget*” pode ser utilizado para buscar (baixar) objetos em um servidor HTTP por meio da linha de comando. Ele permite salvar o documento transferido via HTTP para um arquivo local no computador.

Quadro 6.9 – Resultado do uso do comando WGET

```
1. * wget   http://www.yy.com.br/aulaphp/cadastro.html
2.
3. --2016-10-23 11:35:43--  http://www.yy.com.br/aulaphp/cadastro.html
4. Resolvendo www.yy.com.br (www.yy.com.br)... 127.0.0.1
5. Conectando-se a www.yy.com.br (www.yy.com.br)|127.0.0.1|:80...
   conectado.
6. A requisição HTTP foi enviada, aguardando resposta... 200 OK
7. Tamanho: 943 [text/html]
8. Salvando em: "cadastro.html"
9.
10. 100%[=====] 943 --.-K/s   em 0s
11.
12. 2016-10-23 11:35:43 (54,7 MB/s) - "cadastro.html" salvo [943/943]
```

Fonte: Elaborado pelo autor.

O “*wget*” pode ser utilizado para fazer o download de toda a estrutura do site quando utilizado com a opção de recursividade (percorrendo todos os hiperlinks da página), bastando, para isso, utilizar a opção “-r” (podendo inclusive indicar quantos níveis de hiperlinks deve percorrer).

Quadro 6.10 – Resultado do comando WGET com a opção “-S” para visualização do cabeçalho HTTP

```
1. * wget http://www.yy.com.br/aulaphp/cadastro.html -S
2.
3. --2016-10-23 11:46:20-- http://www.yy.com.br/aulaphp/cadastro.html
4. Resolvendo www.yy.com.br (www.yy.com.br)... 127.0.0.1
5. Conectando-se a www.yy.com.br (www.yy.com.br)|127.0.0.1|:80...
   conectado.
6. A requisição HTTP foi enviada, aguardando resposta...
7. HTTP/1.1 200 OK
8. Date: Sun, 23 Oct 2016 13:46:20 GMT
9. Server: Apache/2.4.7 (Ubuntu)
10. Last-Modified: Mon, 16 May 2016 22:23:39 GMT
11. ETag: "3af-532fd138c3d95"
12. Accept-Ranges: bytes
13. Content-Length: 943
14. Vary: Accept-Encoding
15. Keep-Alive: timeout=5, max=100
16. Connection: Keep-Alive
17. Content-Type: text/html
18. Tamanho: 943 [text/html]
19. Salvando em: "cadastro.html.l"
20.
21. 100%[=====>] 943 --.-K/s   em 0s
22.
23. 2016-10-23 11:46:20 (101 MB/s) - "cadastro.html.l" salvo [943/943]
```

Fonte: Elaborado pelo autor.

Na versão anterior, nas linhas 7 a 17, foram acrescentadas as informações de cabeçalho (HEAD) do HTTP para verificação.

## Síntese

Neste capítulo, estudamos a camada de aplicação usando como referência os protocolos NTP e HTTP. Outros protocolos de aplicação usados na administração das redes já haviam sido apresentados em capítulos anteriores.

O protocolo HTTP foi usado como base para trazer outras tecnologias estudadas neste capítulo e que fazem parte do uso da Internet. Também

verificamos que o protocolo HTTP está sendo utilizado como base para a distribuição de fluxos de vídeos, usando o padrão DASH, e simulamos os comandos básicos do HTTP com os utilitários *ncat*, “*curl*” e “*WireShark*”.

O uso de HTML para a descrição dos conteúdos e as formas de interação com o usuário com a utilização de formulários e linguagens de programação também foram vistos, mas eles serão mais detalhados no próximo capítulo.

## Atividades

1. Além do horário no servidor remoto, qual informação o protocolo NTP deve levar em conta ao sincronizar o tempo do dispositivo local e que pode interferir na interpretação da diferença entre os dois relógios (local e remoto)?
2. Sobre o protocolo HTTP, marque com verdadeiro (V) ou falso (F) nas afirmações a seguir.
  - ( ) A porta padrão para identificação de um serviço HTTP é a 80.
  - ( ) O código de retorno 200 indica que houve erro na recepção da solicitação do cliente no servidor HTTP.
  - ( ) A URL é utilizada apenas nos navegadores para o acesso ao protocolo HTTP.
  - ( ) O CGI é um protocolo utilizado para o envio de informações sobre o servidor HTTP.
3. Qual a diferença entre os comandos GET e HEAD?
4. Qual a função do formulário em uma página HTML?

# 7

## Servidor HTTP e o Desenvolvimento WEB

NESTE CAPÍTULO ESTUDAREMOS na prática as tecnologias do ambiente WEB no servidor, sua configuração, e aprofundaremos sobre como as tecnologias identificadas no capítulo anterior são utilizadas e interagem. Os conceitos das camadas da arquitetura TCP/IP serão aplicados em alguns exemplos práticos, possibilitando o entendimento de como sustentam grande parte dos serviços utilizados na Internet e como a rede de computadores está associada aos cenários verificados no primeiro capítulo. O conhecimento de como funcionam os sistemas WEB auxilia o administrador de redes a entender melhor o impacto que a estrutura sob sua supervisão pode ter no funcionamento dos serviços *online*.

## 7.1 Ambiente WEB

O protocolo HTTP, visto no capítulo anterior, é processado num dispositivo servidor, por meio dos *softwares* que implementam os comandos de requisição e resposta, num processo conhecido como cliente-servidor. O sistema servidor é implementado e parametrizado de maneira a poder filtrar as solicitações para um ou mais domínios específicos, diferentes portas de comunicação e, quando for o caso, executar no lado servidor a automatização das informações (páginas dinâmicas) antes de responder à requisição.

Considerando a documentação disponível na Internet de como o protocolo deve funcionar (RFCs da IETF – *Internet Engineering Task Force*), várias foram as versões de servidores HTTP que puderam ser disponibilizadas para esta tarefa. Alguns sistemas de serviço HTTP se destacam pela quantidade de sites que os utilizam. Este é o caso dos sistemas: apache, Nginx, IIS, Google (NETCRAFT, 2016). Para as práticas de configuração e teste deste capítulo, usaremos o servidor HTTP Apache.

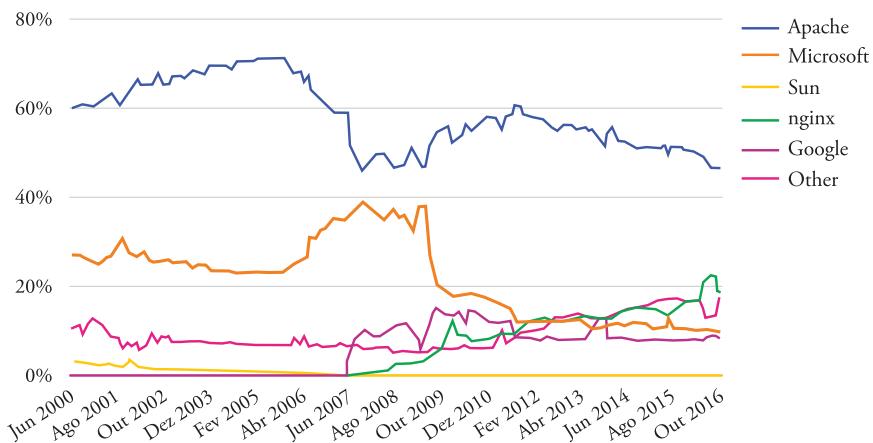
Para a automação das páginas no lado servidor usaremos como referência a linguagem PHP que tem sido utilizada por grandes sites da Internet, como por exemplo o Facebook e Wikipedia (SCHECHTER, 2011; WIKIPEDIA, 2016).

## 7.2 Servidor HTTP APACHE

O servidor HTTP Apache é um projeto *opensource* que vem sendo desenvolvido desde 1995 de maneira colaborativa e conquistou uma grande base de instalações. O processo de desenvolvimento e a metodologia utilizada pelo projeto resultou em um *software* estável e robusto, com uma qualidade comparável à de sistemas proprietários. Seu sucesso provocou a criação de uma fundação que agrupa, desenvolve e mantém dezenas de *softwares open-source* de grande utilização, como por exemplo Tomcat, Hadoop, OpenOffice e Cassandra.

Na figura 7.1, pode ser verificada a constante liderança do uso do sistema Apache como principal opção para servidores HTTP na Internet em mais de duas décadas.

**Figura 7.1 – Uso dos sistemas HTTP na Internet em outubro de 2016, considerando páginas ativas**



Fonte: NETCRAFT (2016).

O servidor Apache pode ser instalado em diferentes sistemas operacionais, é desenvolvido na linguagem C e consegue estender suas funcionalidades por meio de módulos.

O acesso a um servidor HTTP Apache foi verificado no capítulo anterior usando os utilitários *ncat* e *curl*, onde os comandos do HTTP eram respondidos à medida que eram processados pelo servidor. Nas configurações do sistema, verificaremos como habilitar o servidor para responder aos acessos, utilizar outras portas de comunicação além da porta padrão (80) e a execução de uma linguagem adicional (PHP) para o processamento das requisições.

### 7.2.1 Estrutura e arquivos de configuração

Na instalação do servidor Apache num sistema GNU/Linux é comum os arquivos utilizados na configuração do sistema estarem instalados na pasta “/etc/apache2”. O arquivo de configuração principal definido pelo projeto Apache é o “*httpd.conf*”, mas tanto o local quanto o nome do arquivo podem variar conforme a distribuição e o tipo de Unix utilizado. Neste capítulo será utilizada como referência a versão 2.4 do servidor Apache numa distribuição

padrão Debian. Nas distribuições baseadas neste padrão o arquivo de configuração principal é renomeado para “apache2.conf”.

## Saiba mais

Apesar da popularidade do GNU/Linux, este é uma das derivações do sistema Unix. O Unix foi um sistema desenvolvido na década de 1970 com uma perspectiva de portabilidade (deu origem à linguagem C) e foi muito utilizado na área acadêmica para a implementação de várias tecnologias (inclusive um dos pioneiros no uso do TCP/IP). Houve versões proprietárias e *open source*. Outro tipo de Unix que se destaca na Internet pelo desempenho da pilha TCP/IP é o FreeBSD.

Por meio do “apache2.conf” podem ser verificados outros arquivos complementares de configuração, facilitando a administração e manutenção dos serviços em grandes instalações. Nas distribuições GNU/Linux baseadas em Debian, algumas pastas são utilizadas para separar a configuração dos sites ativos (sites-enabled) e configurações complementares do servidor (conf-enabled).

Quadro 7.1 – Formas de inclusão de arquivos de configurações complementares no arquivo apache2.conf

```
Include ports.conf  
IncludeOptional conf-enabled/*.conf  
IncludeOptional sites-enabled/*.conf
```

Fonte: Elaborado pelo autor.

No quadro, tem-se o termo de inclusão (*Include* e *IncludeOptional*), indicando que será lido o arquivo informado ao lado com a sua localização e estes serão interpretados pelo servidor. Na segunda linha, todos os arquivos que tenham a extensão “conf” (\*.conf), que estejam na pasta “conf-enable”, serão incluídos no processo de configuração.

O arquivo “ports.conf” indicado na primeira linha do quadro 7.1, utiliza o termo “Listen” para informar quais portas TCP serão verificadas e capturadas pelo servidor HTTP. Por exemplo, caso tenha sido configurado “Listen 32000” como porta para o servidor Apache, ao acessar determinado site deve ser usada a porta no endereço URL (ver quadro 7.2). Caso não queira que a porta seja associada a todas as interfaces, pode ser limitado para qual IP a porta deve ser capturada, indicando o IP:PORTA, como em “Listen 192.168.0.13:3200”.

Quadro 7.2 – Acesso a site na porta 32000 usando o comando “curl”

```
# curl http://www.yy2.com.br:32000/foto.html

<html>
<head> </head>
<body>
<h1> Texto em HTML</h1>
<p>
Quantos <i><b>Switches</b></i> temos na rede interna que participam <br>
do acesso do sistema administrativo ao banco de dados?<br><br>

</p>
</body>
</html>
```

Fonte: Elaborado pelo autor.

Muitas das configurações feitas para o servidor Apache envolvem a localização dos arquivos, documentos ou códigos no servidor. Estes locais (pastas) não ficam acessíveis diretamente pelo site. Quando um site é configurado e os documentos são copiados para uma determinada pasta (por exemplo, em /var/www/html), o servidor deixa visível para o acesso externo apenas os arquivos ou subpastas que são criadas **a partir** desta pasta inicial. Desta forma, ao acessar através de uma URL, o público externo não consegue acessar a pasta anterior (/var) ou outras pastas do sistema como “/etc”, evitando o acesso a documentos privados ou arquivos de configuração do sistema operacional.

Um site, quando é configurado no servidor, precisa informar pelo menos para qual domínio ele estará associado, qual porta o site publicará o acesso e a pasta no servidor onde estão os documentos (ver quadro 7.3).

#### Quadro 7.3 – Estrutura básica de um site configurado na porta 80

```
<VirtualHost *:80>
    ServerName www.yy.com.br
    DocumentRoot /var/www/yy.com.br
</VirtualHost>
```

Fonte: Elaborado pelo autor.

Neste exemplo acima, o domínio “www.yy.com.br”, ao ser solicitado pelo navegador (e a solicitação chegando no servidor por meio do processo de descoberta de DNS e rotas vistas nos capítulos anteriores), será analisado pelo servidor HTTP o nome de domínio (ServerName) e será direcionado para a pasta “/var/www/yy.com.br” (indicada no DocumentRoot). Também será verificado se a solicitação foi feita para a porta configurada para o site, neste caso a porta TCP 80, padrão do HTTP (<VirtualHost \*:80>). Para o público externo ao servidor, não é permitido acessar documentos que estejam na pasta “/var”. Se tentar fazer um acesso a esta pasta pelo navegador usando o endereço “http://www.yy.com.br/var”, o servidor tentará utilizar uma pasta com o nome “var”, que possa estar **dentro** da pasta “www”, e caso não seja encontrada retornará com um erro.

Outros parâmetros podem ser acrescentados para o site, como permissões de acesso a pastas, separação de registros de acesso, arquivos para registro de erros ou esquemas de autenticação.

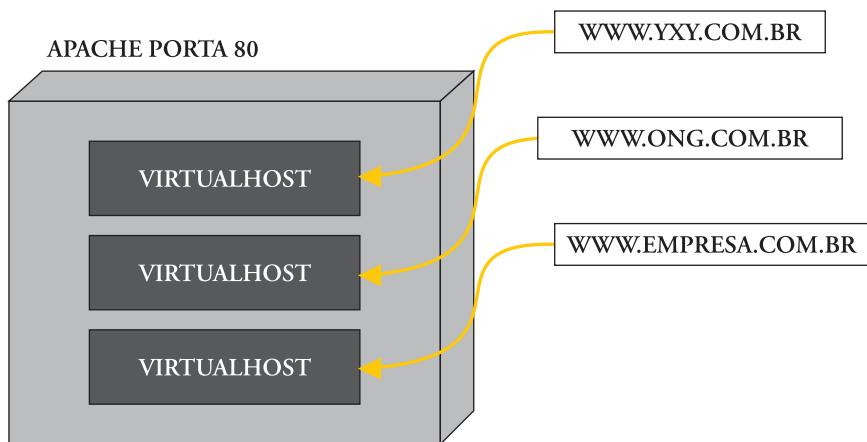
E o serviço HTTP do Apache pode ser configurado para administrar e responder a vários sites no mesmo dispositivo sem a necessidade de instalação de mais um serviço.

### 7.2.2 VirtualHost (Servidores virtuais)

Uma porta TCP ou UDP pode ser usada apenas por um endereço IP de cada vez, como já visto em capítulos anteriores. Quando um aplicativo inicia, ele procura se associar a uma porta do IP que está configurada ao dispositivo. Caso esta porta já esteja utilizada, o sistema operacional emite um erro indicando não ser possível utilizar a porta.

Para possibilitar a implementação de vários sites no mesmo servidor e que, por facilidade de acesso, precisam ser encontrados todos na porta 80, o Apache implementa um mecanismo chamado de *virtualhost* (ou máquina virtual). Não é o mesmo tipo de virtualização feita com a separação de um sistema operacional, mas permite utilizar a mesma porta para mais de um domínio: o Apache recebe a solicitação para a porta 80, identifica dentro do pacote HTTP o domínio para o qual a requisição foi feita e redireciona para o site correto.

Figura 7.2 – Diferentes domínios no mesmo servidor



Fonte: Elaborado pelo autor.

Na figura 7.2, o diagrama mostra a relação de cada domínio com uma configuração de "<VirtualHost>" específica (ver quadro 7.4), que pode estar em arquivos de configuração separados na pasta "*sites-enabled*". No quadro 7.4 podem ser verificadas duas configurações de sites, cada um com uma porta de acesso, domínio e localização dos arquivos em pastas diferentes.

Quadro 7.4 – Configuração de sites com portas diferentes

<pre>&lt;VirtualHost *:80&gt; ServerName www.yy.com.br DocumentRoot /var/www/yy.com.br &lt;/VirtualHost&gt;</pre>	<pre>&lt;VirtualHost *:3200&gt; ServerName www.empressa.com.br DocumentRoot /var/www/pastaempresa &lt;/VirtualHost&gt;</pre>
---	--

Fonte: Elaborado pelo autor.

### 7.2.3 Módulos complementares

O Apache permite estender suas funcionalidades através de programas independentes, que o administrador do serviço pode habilitar durante a instalação ou posteriormente. Com estes módulos, interpretadores e compiladores de linguagens de programação puderam ser incorporados ao processamento das requisições permitindo um maior desempenho do que o obtido com a execução independente usando o CGI.

Além das linguagens de programação, podem ser desenvolvidas funções para o processamento de protocolo, redirecionamento de URL, autenticação de acesso, monitoramento de conexões, criptografia e outras. Dois módulos não relacionados a linguagem de programação são exemplificados a seguir.

#### 7.2.3.1 Apache módulo “mod\_http2”

Este módulo do Apache implementa as características do protocolo HTTP2 para o processamento de conexões com a nova versão do protocolo HTTP.

Deve ser considerado um aumento do uso de memória para a administração do fluxo de dados entre o servidor e o cliente, que para acontecer necessita do armazenamento de informações do estado das conexões. Este mecanismo (visto no capítulo anterior) foi implementado como uma melhoria nesta nova versão de protocolo e permite que várias transmissões sejam feitas numa mesma sessão de comunicação.

#### 7.2.3.2 Apache módulo “mod\_rewrite”

Este módulo permite reescrever a URL recebida pelo servidor Apache, podendo redirecionar para um caminho no sistema de arquivos ou outra URL. Esta reescrita pode deixar a URL mais simples (URLs amigáveis) que depois é transformada para a URL real (mais longa) e também pode fazer redirecionamentos de emergência ou temporários. Para ativar o módulo, pode ser utilizado o comando “*a2enmod*” indicando o módulo a ser habilitado (ver quadro 7.5).

Quadro 7.5 – Ativação do módulo no Apache usando “a2enmod”

```
# a2enmod rewrite.load
Enabling module rewrite.
To activate the new configuration, you need to run:
service apache2 restart

# service apache2 restart
```

Fonte: Elaborado pelo autor.

Para utilizar a reescrita da URL deve ser indicado na configuração do site que a reescrita está ligada (“on”) e inserir a opção “*FollowSymLinks*”, como mostra o quadro 7.6.

Quadro 7.6 – Configuração da reescrita de URL no arquivo de configuração do site

```
<Directory "/var/www/">
    RewriteEngine on
    Options FollowSymLinks
</Directory>
```

Fonte: Elaborado pelo autor.

No processo de reescrita podem ser utilizadas as informações de cabeçalho do HTTP para criar condições para a mudança da URL. No quadro 7.7, dois exemplos mostram as possibilidades de redirecionamento. Na primeira parte, na linha 5, é usada a variável do HTTP que contém as informações do agente que solicitou a requisição para o servidor. Neste caso, se contiver a palavra “android” (como acontece quando um dispositivo com sistema *Android* acessa um site) ele redirecionará a requisição (linha 6) para uma página específica chamada “index-android.html”. O “NC” ao final indica que não será diferenciado se a *string* tem maiúsculas ou minúsculas.

Quadro 7.7 – Exemplo de reescrita de URL

```
1. <Directory "/var/www/">
2.   RewriteEngine on
3.   Options FollowSymLinks
4.
5.   RewriteCond "%{HTTP_USER_AGENT}" "(android)" [NC]
6.   RewriteRule ".*" "index-android.html" [L]
7.
8.   RewriteCond "%{REMOTE_ADDR}" "192.168.0.13"
9.   RewriteRule "^foto\.html$" "teste.html" [L]
10.  </directory>
```

Fonte: Elaborado pelo autor.

No segundo exemplo do quadro 7.7 é feito um teste se o endereço remoto é “192.168.0.13” (linha 8) e se foi solicitado o arquivo “foto.html” (linha 9). Se as **duas** condições foram atendidas, a requisição é redirecionada pelo servidor Apache para o arquivo “teste.html”. Ou seja, mesmo se o usuário solicitar explicitamente o arquivo “foto.html” a partir do dispositivo “192.168.0.13”, o servidor não retornará este arquivo e sim o novo documento “teste.html”.

## 7.3 Desenvolvimento WEB no lado Servidor

A utilização de uma linguagem de programação para processar parte da solicitação HTTP no servidor proporcionou uma alternativa para o desenvolvimento de sistemas de informação, uma vez que o sistema operacional onde seria executado o código final deixa de ser uma preocupação. Bastaria ter um navegador no cliente (independente do sistema operacional e do *hardware*) e o sistema ficaria no servidor, respondendo a todas as solicitações por meio de um conteúdo formatado em HTML. Se as informações são buscadas num banco de dados, em arquivos ou outros servidores, isso fica oculto para o usuário final.

Como foi verificado no capítulo 6, o CGI tornou possível esta condição de processamento da requisição via linguagem de programação.

### 7.3.1 Uso do CGI

Para testar a criação de páginas dinâmicas por meio da interface CGI pode ser utilizada a própria linguagem de *scripts* da linha de comando de um

sistema padrão Unix. Na linha de comando até o momento foram apenas executados os comandos e ferramentas instaladas no sistema. Mas podem ser criadas variáveis, estruturas de repetição e condição para automatizar uma série de tarefas diretamente na linha de comando ou agrupadas em arquivos executáveis.

## Saiba mais

Neste caso, usaremos algumas estruturas simples para criar páginas dinâmicas de exemplo, por meio da linguagem de script da linha de comando (*shell*). Mas esta linguagem possui capacidade para criar complexas ferramentas para o auxílio na administração de servidores. Podem ser pesquisados na Internet exemplos de uso deste tipo de programa, sendo o BASH o tipo de *shell* mais popular.

Primeiramente, deve ser habilitado na configuração do site o uso do CGI (ver quadro 7.8). A primeira parte da linha tem a indicação de que será habilitado o uso de *scripts* (*ScriptAlias*), por meio do uso do caminho “*/cgi-bin/*” na URL e executará o script que esteja na pasta “*/var/www/yy.com.br/cgi-bin/*”.

Quadro 7.8 – Configuração para o uso de *scripts* CGI no site

```
ScriptAlias /cgi-bin/ /var/www/yy.com.br/cgi-bin/
```

Fonte: Elaborado pelo autor.

As variáveis principais que podem ser acessadas via interface CGI estão listadas no quadro 7.9. Estas variáveis estão relacionadas a informações transmitidas no cabeçalho do HTTP e podem ser acessadas pelo programa executado pelo Apache.

Quadro 7.9 – Lista de variáveis usadas no CGI

Variáveis CGI	Descrição
“AUTH_TYPE”	Tipo de autenticação usada para o acesso a páginas do servidor.

Variáveis CGI	Descrição
“CONTENT_LENGTH”	O tamanho da mensagem enviada em bytes.
“CONTENT_TYPE”	Tipo de conteúdo transmitido (visto no capítulo anterior).
“GATEWAY_INTERFACE”	Versão do CGI.
“PATH_INFO”	Caminho onde está o CGI.
“PATH_TRANSLATED”	Caminho físico (sistema de arquivos) onde está o CGI (usa o PATH_INFO).
“QUERY_STRING”	Onde ficam registradas as variáveis de formulário passadas para o servidor.
“REMOTE_ADDR”	Endereço IP do cliente.
“REMOTE_HOST”	Nome do dispositivo do cliente.
“REMOTE_IDENT”	Pode ser utilizado para identificar a origem da conexão (nem sempre usado).
“REMOTE_USER”	Identifica o usuário, caso tenha sido indicada uma autenticação.
“REQUEST_METHOD”	Palavra-chave (GET, PUT, HEAD, PUT, DELETE) indicando o comando HTTP usado na transferência.
“SCRIPT_NAME”	Nome do CGI a ser executado.
“SERVER_NAME”	IP ou nome do servidor.
“SERVER_PORT”	Porta utilizado do servidor.
“SERVER_PROTOCOL”	Versão do HTTP.
“SERVER_SOFTWARE”	Nome e versão do sistema que implementa o HTTP.

Fonte: Elaborado pelo autor com base em RFC3875 (2016).

No primeiro exemplo de uso do CGI (quadro 7.10), foi apenas impressa (usando comando “echo”) a informação vinda por meio das variáveis do HTTP (linhas 7, 8 e 9) que o servidor recebeu na requisição. Estas informações podem ser usadas juntamente com os conteúdos e são enviadas pelas conexões HTTP feitas por meio de formulários e dos agentes usados no cliente.

Quadro 7.10 – Exemplo de CGI com o uso de *shell* (Bash)

```

1. #!/bin/bash
2. echo "Content-type: text/html"
3. echo
4. echo "<html><body>"
5. echo "<h1>Mensagem de texto no servidor web</h1>"
6. echo "<hr>Variáveis do CGI<br><pre>"
7. echo "ENDEREÇO REMOTO (BROWSER) = $REMOTE_ADDR"
8. echo "FORMULARIO (GET) = $QUERY_STRING"
9. echo "TAMANHO = $CONTENT_LENGTH"
10. echo "</pre><hr>"
11. echo "No caso de POST precisa usar read do shell <br>"
12. read form
13. echo $form
14. echo "</body></html>"
```

Fonte: Elaborado pelo autor.

O exemplo seguinte (quadro 7.11) mostrará no navegador do cliente quais são os **aniversariantes do dia** automaticamente. Este exemplo está separado em **três** partes: a parte de cima é um arquivo texto com informações de aniversariantes (dia e nome), a parte do meio é o *script* que fará a verificação do aniversariante do dia e a última parte é o resultado que será enviado para o navegador do usuário. O arquivo texto que contém os aniversariantes tem o nome indicando o mês dos aniversariantes: no caso do exemplo, o nome é “base” + mês novembro + “.dat” = “base11.dat”.

No *script* (parte do meio do quadro) é feito o uso de funções básicas da linha de comando do GNU/Linux, como a data local (separada por dia e mês nas linhas 4 e 5) que são colocadas nas variáveis “mes” e “dia”. Também é utilizado um comando de pesquisa de texto chamado “grep” (linha 9), que pesquisa uma *string* dentro de um arquivo.

Quadro 7.11 – Página HTML com os aniversariantes do dia com o uso de *shell* (Bash)

```
1. # cat base11.dat
2. 05 BRUNO SILVA
3. 05 PEDRO COSTA
4. 10 YANKO SOUZA
5. 12 RAFAEL JUNIOR
6. 16 SILVA JUNIOR
7. 31 MARIA ALBUQUERQUE
```

```
1.#!/bin/bash
2.echo "Content-type: text/html"
3.echo ""
4.mes=$( date +%m )
5.dia=$( date +%d )
6.echo "<h1> Aniversariantes do dia</h1>"
7.echo "<hr>"
8.echo "<h3> Hoje, os aniversariantes sao:</h3><pre>"
9.grep $dia base$mes.dat
10.echo "</pre>"
```

```
1. Content-type: text/html
2.
3. <h1> Aniversariantes do dia</h1>
4. <hr>
5. <h3> Hoje, dia<u> 31 </u>, os aniversariantes sao:</h3>
6. 31 MARIA ALBUQUERQUE
```

Fonte: Elaborado pelo autor.

Considerando que seja o dia **31/11**, quando o cliente acessa o CGI de aniversários (<http://www.teste.com.br/cgi-bin/aniversario.sh>), o servidor irá executar o arquivo do CGI. Este programa irá pegar o mês (11) e o dia (31) e colocar nas variáveis (linhas 4 e 5). Depois, na linha 9, o programa executa o comando “grep”, que usará a variável “\$dia” (que contém 31) para pesquisar no arquivo “base11.dat” (a variável \$mes foi trocada por 11) e mostrará as linhas do arquivo (primeira parte) que contêm o número 31. No exemplo, ao final mostra o resultado do CGI com o texto “31 MARIA ALBUQUER-

QUE” no resultado dos aniversariantes do dia (caso tenha mais de um, serão mostradas todas as linhas que possuem o mesmo número.

### 7.3.2 PHP

Enquanto o uso de *scripts* de linha de comando pode automatizar algumas situações, outras linguagens podem trazer mais recursos no processamento das páginas dinâmicas, envolvendo o uso de banco de dados e melhorando o desempenho da aplicação. O PHP é uma das linguagens mais utilizadas na Internet (PHPNETCRAFT, 2013) e será utilizada como referência.

No capítulo passado, visualizamos o retorno de um formulário por meio de solicitação via comando “curl”. Neste capítulo será verificada a criação dinâmica de formulários, respostas e o uso de banco de dados para fornecer conteúdo para o cliente por meio do uso do HTTP. No quadro 7.12, podemos verificar um formulário HTML sendo mostrado para o usuário pelo navegador, onde os itens a serem selecionados (*checkbox*) foram transmitidos por HTTP e foram fornecidos por uma base de dados no servidor.

Apesar de alguns exemplos em PHP e Javascript envolverem um conhecimento básico de lógica de programação, podem ser acompanhadas as explicações de cada quadro de exemplo para entender o funcionamento geral, sem entrar em detalhes específicos das linguagens de programação.

Quadro 7.12 – Exemplo de formulário construído com PHP e banco de dados

The screenshot shows a web page with a light gray background. At the top center, the title "Relatorio de Vendas por Produto" is displayed in bold black font. Below the title, there is a label "Selecione o produto:" followed by a horizontal line for input. To the left of the input field is a small square checkbox. To the right of the input field is a list of three items: "FURADEIRA", "NOTEBOOK", and "XICARA". Each item has a small square checkbox next to it. At the bottom of the form, there is a single word "Envia" enclosed in a rectangular button.

```
1. <h3>Relatorio de Vendas por Produto</h3>
2. <form name="CAD" action="lista-vendas.php" method="POST">
3.   <fieldset>
4.     <legend>Selecione o produto:</legend>
5.
6.   <?php
7.   //informacoes do banco
8.   $svr = "192.168.0.1";
9.   $user = "sistema";
10.  $password = "123456";
11.  $base = "comercial";
12.
13. // Estabelece a conexão com o servidor
14. $banco = new mysqli($svr, $user, $password, $base);
15.
16. //verifica se a conexão completou com sucesso
17. if ($banco->connect_error) {
18.   die("Erro na conexão: " . $banco->connect_error);
19. }
20. $sql = "SELECT codproduto, dsproduto FROM produto ";
21. $rs = $banco->query( $sql );
22.

23. if ($rs->num_rows > 0) {
24.   // imprime todas as linhas da tabela
25.   while($registro = $rs->fetch_assoc()) {
26.
27.     $cod = $registro["codproduto"];
28.     $dsprod = $registro["dsproduto"];
29.
30.     echo "<input type='checkbox' name='prod[]' value='".$cod."'>".$dsprod."<br>";
31.   }
32. } else {
33.   echo "A tabela esta vazia!!!!";
34. }
35. $banco->close();
36. ?>
37. </fieldset>
38. <input type="submit" name="Envia" value="Envia">
39. </form>
```

```

1. <html>   <head><title>Form - 1</title></head>
2. <body>
3. <h3>Relatorio de Vendas por Produto</h3>
4. <form name="CAD" action="lista-vendas.php" method="POST">
5.   <fieldset>
6.     <legend>Selecione o produto:</legend>
7.     <input type='checkbox' name='prod[]' value='1'>FURADEIRA<br>
8.     <input type='checkbox' name='prod[]' value='2'>NOTEBOOK<br>
9.     <input type='checkbox' name='prod[]' value='3'>XICARA<br>
10.   </fieldset>
11.
12. <input type="submit" name="Envia" value="Envia">
13. </form>
14. </body>
15. </html>

```

Fonte: Elaborado pelo autor.

O exemplo do quadro 7.12 é mostrado em três partes: na primeira, o resultado do HTML formatado pelo navegador; na segunda parte, a listagem do código PHP, que permite criar o formulário; e na terceira o HTML de resultado, que será mostrado pelo navegador. Ao verificar a lista do programa na segunda parte do exemplo, pode-se identificar um início de TAGs HTML da linha 1 a 4. Na sequência, a indicação do início do código em PHP (linha 6) e a inicialização de variáveis que serão usadas para a conexão ao banco de dados (linhas 8 a 11) e a conexão propriamente dita (linha 14). Nas linhas 20 e 21 temos a seleção no banco de dados dos produtos cadastrados. A descrição destes produtos (“*dsproduto*”) que foi inserida na variável “\$dsprod”, será mostrada no formato de um INPUT na linha 30.

Com esta lógica, em vez de criar um HTML contendo a descrição **estática** dos produtos a serem listados no formulário, os produtos podem ser alterados dinamicamente com a inserção, alteração ou eliminação de algum destes itens no banco de dados. O programa não se altera caso estes itens sofram manutenção, e o formulário permanece sempre atualizado com a última versão dos produtos.

Ao enviar a solicitação do relatório para o servidor, indicando os produtos a serem incluídos na listagem, será executado o código do quadro 7.13. Este relatório recebe por meio do método POST do HTTP as informações de quais itens do “checkbox” foram selecionados (ver primeira parte do quadro 7.12) e executa uma pesquisa no banco de dados com cada um destes itens.

O resultado pode ser visualizado na primeira parte do exemplo do quadro 7.13. Para cada produto selecionado anteriormente (FURADEIRA=1 e NOTEBOOK=2) foram selecionadas as vendas e apresentadas em uma tabela HTML.

Quadro 7.13 – Listagem de vendas por produto construída com PHP e banco de dados

## Listando as vendas do produto !!

Data:30/Oct

Produto:1

Data Venda	Quant	V. Unit.	Total
2015-11-16	55	2.5	137.5
2015-07-04	20	2.5	50

Produto:2

Data Venda	Quant	V. Unit.	Total
2015-11-09	10	15	150
2015-07-04	30	15	450
2015-12-21	5	15	75
2015-12-01	6	15	90

```

1. if (isset($_POST["prod"])) {
2.
3. foreach ($_POST["prod"] as $codproduto) {
4.
5. $sql = "SELECT codproduto, valorunit, qtdvenda, dtvenda FROM
       vendas WHERE codproduto = $codproduto";
6. $rs = $banco->query( $sql );
7.
8. if ($rs->num_rows > 0) {
9.
10. echo "<br>Produto:". $codproduto;
11. echo "<table border=1>";
12. echo "<tr><td>Data Venda</td><td>Quant</td> <td> V. Unit.
       </td><td>Total</td></tr>";
13. // imprime todas as linhas da tabela
14. while($registro = $rs->fetch_assoc()) {
15.   $qtd    = $registro["qtdvenda"];
16.   $preco   = $registro["valorunit"];
17.   $dtvenda = $registro["dtvenda"];
18.   echo "<tr><td>". $dtvenda .</td><td> ". $qtd. " </td><td> ".
       $preco. " </td><td> ". $qtd*$preco. "</td></tr>";
19.
20. }
21. echo "</table>";
22. } else {
23.   echo "A tabela esta vazia!!!!";
24. } } }
```

```

1. <h3> Listando as vendas do produto !!</h3>
2. Data:01/Nov<BR>
3. <br>Produto:1
4. <table border=1>
5. <tr><td>Data Venda</td><td>Quant</td><td>V.
       Unit.</td><td>Total</td></tr>
6. <tr><td>2015-11-16</td><td>55</td><td>2.5</td><td>137.5</td></tr>
7. <tr><td>2015-07-04</td><td>20</td><td>2.5</td><td>50</td></tr>
8. </table>
9. <br>Produto:2
10. <table border=1>
11. <tr><td>Data Venda</td><td>Quant</td><td>V. Unit.</td><td>Total</td>
     </tr>
12. <tr><td>2015-11-09</td><td>10</td><td>15</td><td>150</td></tr>
13. <tr><td>2015-07-04</td><td>30</td><td>15</td><td>450</td></tr>
14. <tr><td>2015-12-21</td><td>5</td><td>15</td><td>75</td></tr>
15. <tr><td>2015-12-01</td><td>6</td><td>15</td><td>90</td></tr>
16. </table></body>
```

Fonte: Elaborado pelo autor.

Na segunda parte do exemplo, a linha 3 indica que será executado o trecho do programa seguinte (linhas 5 a 23) para cada produto da lista recebida pelo formulário HTML. Cada número de produto é pesquisado no banco de dados (linha 5) e o resultado de todos os registros das vendas deste produto é listado em um formato de tabela do HTML (linhas 11 a 21).

Para o navegador do cliente, apenas são transmitidas as informações resultantes do programa PHP no formato HTML (como pode ser verificado na terceira parte do exemplo). O navegador não recebe o código PHP, portanto, no lado cliente não é possível verificar qual parte é fixa no HTML e qual parte é dinamicamente inserida por meio do algoritmo e do banco de dados.

Este tipo de construção possibilitou o desenvolvimento de muitas aplicações no lado servidor que podem ser hospedadas fora da empresa, que antes eram executadas apenas em rede local ou em aplicativos isolados. Muitos serviços estão disponíveis apenas no formato WEB, criando uma dependência cada vez maior da rede de computadores e da Internet, tanto para ambientes corporativos quanto para ambientes residenciais.

Muitas empresas baseiam sua prestação de serviços apenas no ambiente virtual da Internet e sem a rede não poderiam existir.

## 7.4 Desenvolvimento WEB no lado cliente

No capítulo anterior foi identificada a situação do HTML como sendo a tecnologia a ser percebida no lado cliente durante a transmissão dos pacotes HTTP. Mas, juntamente com o HTML foram desenvolvidas algumas tecnologias auxiliares e complementares. O CSS (*Cascade Style Sheet*) é uma forma de trabalhar a aparência dos documentos HTML. Cores, imagens, fontes de letra, espaçamentos e bordas são manipulados

com maior precisão e com a possibilidade de padronização entre as diversas páginas de conteúdo.

Para trabalhar a interação e navegação das páginas, tem sido usada como padrão a linguagem de *script* chamada popularmente de JavaScript.

### 7.4.1 JavaScript

Esta linguagem foi criada pela Netscape em 1995 e depois padronizada internacionalmente como ECMAScript. Ela foi incorporada nos navegadores se tornou uma linguagem que auxilia a manipulação de objetos HTML e CSS, permitindo mudanças dinâmicas no conteúdo.

A evolução e a capacidade do JavaScript têm levado à criação de diversos produtos (bibliotecas) que podem ser usados como base de aplicações maiores e que permitem que aplicações envolvendo a manipulação de imagens, vetores gráficos e dados de servidores sejam processados no navegador.

Serão apresentados dois exemplos de JavaScript para demonstrar o uso da linguagem. O primeiro exemplo mostrará o uso do JavaScript para a consistência das informações de preenchimento de formulários. O segundo exemplo mostrará de maneira simples como um *script* pode ser usado para manipulação de TAGs HTML numa animação de um objeto.

Quadro 7.14 – Exemplo de JavaScript fazendo a consistência de um formulário

```
1. <body>
2. <form id="form1" >
3. Nome: <input type="text" id="nome" name="nome"><BR>
4. Endereço: <input type="text" id="end" name="endereco"><BR>
5. <input type="Button" value="Enviar" onclick="teste_form();">
6. </form>
7.
8. <div id="area_de_erro"></div>
```

```
1. <script language="JavaScript">
2.
3. function teste_form() {
4.
5.     var obj_form = document.getElementById('form1');
6.
7.     var campo_nome = document.getElementById('nome');
8.     var campo_end = document.getElementById('end');
9.
10.    var obj_msg_erro = document.getElementById('area_de_erro');
11.    var msg_erro = '';
12.
13.    if(campo_nome.value == '')
14.        msg_erro = 'campo NOME vazio';
15.    else if(campo_end.value == '')
16.        msg_erro = 'campo ENDEREÇO vazio';
17.
18.    if(msg_erro == '')
19.        obj_form.submit();
20.    else
21.        obj_msg_erro.innerHTML = msg_erro;
22. }
23. </script>
```

Fonte: Elaborado pelo autor.

O exemplo acima (quadro 7.14) está separado em duas partes: na primeira é mostrado um trecho de formulário HTML, e na segunda o JavaScript que será acionado pelo formulário antes do envio ao servidor, para conferir o que foi preenchido.

Nas linhas 3 e 4 da primeira parte são utilizadas as TAGs INPUT para coletar as informações digitadas pelo usuário e é inserida a identificação de cada entrada (“id”). Ao finalizar a digitação, o usuário clica no botão que executa o JavaScript (“onclick=teste\_form()”).

Na segunda parte do exemplo, o código JavaScript é executado e inicia colocando em variáveis os itens identificados do formulário (linhas 5 a 10).

A linha 13 mostra um exemplo do teste para verificar se foi digitado algo ou o campo está vazio: o conteúdo do “campo\_nome” é comparado com uma string vazia. Se estiver vazia é colocada uma mensagem de erro na variável “msg\_erro”. Se não estiver vazia, testa o outro campo.

Na linha 18, caso tenha alguma informação na variável de erro, o script insere a mensagem na área destinada ao erro no HTML. Se não tiver

nenhuma informação na variável de erro é porque o usuário digitou todas as informações e o formulário pode ser enviado (linha 19).

No segundo exemplo (quadro 7.15), um objeto será movimentado dentro da área do navegador de 20 em 20 milissegundos. Na linha 4 é indicado que será usado o método de posicionamento “*absolute*”, que permite a alteração da posição, e nas linhas 11 a 13 é criado o objeto que será movimentado.

Na linha 11 é feita a identificação do objeto que será movimentado com o nome de “flecha”.

Quadro 7.15 – Exemplo movimento de objeto no navegador usando JavaScript

```

1. <head>
2. <style type="text/css">
3. DIV {
4.     position: absolute;
5. }
6. </style>
7. </head>
8.
9. <body>
10.
11. <div id="flecha">
12.     <p>--> </p>
13. </div>
14.
15. <script language="JavaScript">
16.     var x = 0;
17.
18.     function anda() {
19.         var flecha = document.getElementById("flecha");
20.         if (x < 250)
21.             x++;
22.         else
23.             x=0;
24.         flecha.style.left = x;
25.     }
26.
27.     setInterval(function() { anda() }, 20);
28.
29. </script>
30. </body>
```

Fonte: Elaborado pelo autor.

Entre as linhas 16 e 25 está o algoritmo que movimenta o objeto “flecha” pela tela. Na linha 16 é criada uma variável que vai conter o valor da

posição na horizontal. As linhas 20 e 21 testam se a quantidade de movimentos chegou ao limite (aos 250 pixels da tela) e se ainda não chegou, aumenta mais um ponto no movimento (“`x++`”). A linha 24 leva o valor acumulado da variável “`x`” para a margem esquerda do objeto (“`flecha`”). Cada vez que a função “`anda()`” é executada aumenta a margem esquerda da “`flecha`” até chegar em 250.

Na linha 27 indica como será feito o movimento: será executada a função “`anda()`” a cada 20 milissegundos.

## Saiba mais

Apesar de ter sido criado para uso em navegadores (*browsers*), o Javascript tem ganhado espaço também no lado servidor. O ambiente Node.js permite que sejam criadas aplicações inteiras usando Javascript. A capacidade de processar uma grande quantidade de conexões concorrentes tem levado alguns sites a utilizar o Node.js como ambiente de processamento de suas requisições via rede.

### 7.4.2 Ajax (Asynchronous JavaScript e XML)

Muitos sistemas WEB têm a necessidade de atualização de alguma das informações da página de maneira mais frequente que outras. Nestes casos, o envio de solicitação de atualização das informações e a reescrita de toda a página, como foi idealizado inicialmente, acaba impactando no desempenho do sistema.

Para permitir o acesso ao servidor sem o impacto de reescrever toda a página, como no caso de um *link* ou a submissão de um formulário, foi criada a tecnologia chamada Ajax.

O Ajax permite que o Javascript possa enviar uma solicitação ao servidor em paralelo com a operação normal da página no sistema. Com isso, algumas informações podem ser atualizadas em partes específicas do site, enquanto outras podem aguardar a atualização do sistema via requisição padrão.

No exemplo do quadro 7.16 temos uma das formas de execução da solicitação ao sistema usando o objeto “*XMLHttpRequest()*”. Este objeto faz uma conexão usando o protocolo HTTP até o servidor e recebe uma resposta diretamente em uma variável (“*ajax.responseText*”). O exemplo está separado em três partes: na primeira temos o HTML, que executará a função Javascript; na segunda parte, é listado o Javascript que fará a transmissão do HTTP até o servidor; e na terceira o programa PHP receberá a conexão e enviará uma resposta diretamente ao JavaScript.

Algumas características que foram verificadas no capítulo anterior sobre o funcionamento do HTTP são utilizadas neste exemplo: os códigos de retorno do HTTP, os métodos de solicitação (GET e POST), os tipos de conteúdo (“Content-Type”).

Quadro 7.16 – Exemplo de execução de código AJAX

```
1. <form name="cadastro" method="post" action="">
2. Nome: <input name="solicitante" type="text" id="nome" value="">
3. <br>
4. Valor: <input name="valor" type="text" id="valor" value="">
5. <br><button onclick="executa(); return false;"> Calcular
   </button>
6. </form>
7.
8. <br>Resultado:
9. <p id="resultado"> </p>
```

```
1. <script>
2. var ajax;
3. function executa() {
4.
5.     ajax = new XMLHttpRequest();
6.
7.     if (ajax) {
8.
9.         nome = document.getElementById("solicitante").value;
10.        valor = document.getElementById("valor").value;
11.        query = "nome=" + nome + "&valor=" + valor;
12.
13.        ajax.open("POST", "/ajax/executa.php");
14.        ajax.setRequestHeader("Content-Type", "application/x-www-
form-urlencoded");
15.        ajax.send(query);
16.
17.        ajax.onreadystatechange =
18.            function () {
19.                if (ajax.readyState == 4) {
20.                    //200=ok, 404=not found
21.                    if (ajax.status == 200) {
22.                        document.getElementById("resultado").innerHTML =
23.                            ajax.responseText;
24.                    } else {
25.                        document.getElementById("resultado").innerHTML =
26.                            ajax.statusText;
27.                    }
28.                }
29.            }
30.
31. </script>
```

```
1. <?php
2. //Executa atraves de chamada ajax
3.
4. $nome = $_POST["nome"];
5. $num = $_POST["valor"];
6.
7. $resultado = $num * $num * $num;
8.
9. echo "Caro $nome, o cubo de $num e $resultado";
10. ?>
```

Fonte: Elaborado pelo autor.

Ao digitar o nome do solicitante e o valor no formulário, o usuário clica no botão que executa o JavaScript no navegador e este por sua vez faz um acesso ao servidor, enviando os dados que o permitirão executar um cálculo e enviar uma resposta. Considerando que o próprio JavaScript poderia calcular o cubo de um número sem a participação do servidor, este exemplo serve para apresentar o mecanismo de funcionamento do Ajax sem criar uma maior complexidade no processo.

Na linha 5 da listagem do JavaScript está a inicialização do Ajax e nas linhas 9 a 11 as informações dos campos do formulário são organizadas para serem transmitidas.

Na linha 13 é iniciada a conexão HTTP indicando o POST como o método de envio, o programa “executa.php” como destino das informações e na linha 14 é definido o tipo de conteúdo como formatado para formulário.

Ao serem enviadas as informações via POST, a página não é redesenhada e o JavaScript aguarda o retorno do servidor, sem interromper ou suspender outras funcionalidades que estejam em operação na página.

Quando a resposta do servidor é recebida pelo JavaScript, é ligado um sinalizador. Na linha 17 com o uso do “*onreadystatechange*”, quando esta variável recebe o valor 4, indica que o resultado chegou. Neste caso, na linha 21 é verificado se o código de retorno do HTTP é 200 (ver capítulo 6, seção 6.4.7). Se for o caso, indica que o recebimento foi finalizado com sucesso e o resultado da resposta (“*responseText*”) é inserido na área de resposta chamada “resultado” no HTML. Se o processamento teve problemas, o código de retorno é inserido na área de respostas.

Na terceira parte do exemplo, o programa PHP apenas recebe as informações, calcula o cubo do número enviado e retorna uma *string* com todas as informações (linha 9).

## Síntese

Neste sétimo capítulo, com a verificação dos procedimentos de configuração do servidor HTTP Apache e os exemplos de como um sistema WEB se comporta, considerando tanto o processamento do lado servidor quanto o

processamento do lado cliente, foi possível identificar como as redes sustentam as trocas de informações entre os dispositivos. Alguns exemplos de como são automatizados os acessos a bases de dados e como as respostas dos servidores podem ser condicionadas aos parâmetros informados pelos usuários em formulários, contribuem para a identificação de como as tecnologias estão inter-relacionadas. Para processar uma requisição do lado cliente, podem ser incluídos vários servidores (banco de dados, sistema HTTP, sistema de autenticação, sistema de criptografia, sistema de arquivos) em endereços diferentes, com os pacotes sendo roteados entre cada rede e cada dispositivo da estrutura.

## Atividades

1. Marque com V (verdadeiro) e F (falso) as afirmações a seguir sobre o servidor HTTP Apache.
  - ( ) Pode ser configurado para atender a portas TCP diferentes da porta padrão 80, mas para isso precisa configurar IPs diferentes.
  - ( ) O servidor Apache pode ser configurado para processar a resposta de uma requisição por meio de uma linguagem de programação.
  - ( ) Para usar um módulo no servidor Apache é necessário primeiro desenvolver um *script* Javascript, como mostrado nos exemplos.
  - ( ) Por meio das variáveis do servidor HTTP disponíveis para o CGI, é possível identificar o endereço IP do dispositivo cliente.
2. Em um dos exemplos de reescrita de URL no servidor Apache foi utilizada a variável “HTTP\_USER\_AGENTE” do protocolo HTTP. Qual o objetivo do uso desta variável no exemplo?
3. Qual a vantagem do uso de PHP para construir os itens de um formulário a partir do acesso a um banco de dados?
4. O Ajax utiliza o protocolo HTTP para fazer o acesso ao servidor. Qual a vantagem do uso do Ajax, considerando que o formulário também faz a transmissão das informações por meio do HTTP?

# 8

## Transferência e Compartilhamento de Arquivos

NESTE CAPÍTULO ESTUDAREMOS os protocolos e sistemas relacionados à transferência de arquivos entre equipamentos e o compartilhamento de arquivos, pastas e discos por meio da rede local e da Internet. Serão verificados os protocolos utilizados para estas trocas e como estas necessidades de transferência de arquivos entre equipamentos iniciou, e entender como as redes locais estabeleceram formas de tratar esta questão. Será abordado o uso de armazenagem de arquivos em nuvem e utilitários que podem auxiliar nas transferências e cópias de segurança utilizadas rotineiramente nos servidores.

## 8.1 Arquivos

Muitas aplicações têm formas e sequências específicas para a gravação dos dados que precisam ser armazenados em computadores para posterior recuperação. Para este conjunto de dados chamamos de **arquivo**, e apesar de todos serem uma sequência de bytes, cada sequência pode significar diferentes conteúdos (vídeos, música, texto). Este arquivo é gravado em uma estrutura que permite que a informação seja encontrada, identificada e tenha dados sobre a validade, tamanho e permissão de acesso associada a este grupo de bytes. Esta estrutura chamamos de **sistemas de arquivos**.

Enquanto um arquivo é criado em um dispositivo, muitas vezes ele precisa ser verificado ou alterado em outro dispositivo. No início do uso de microcomputadores nas empresas, os arquivos podiam ser transferidos fisicamente usando meios magnéticos como transporte, e, ao se disseminar o uso das redes de computadores, uma nova ótica se estabeleceu com relação à transferência destas informações.

Neste capítulo verificaremos duas formas de possibilitar o uso do arquivo em outro dispositivo por meio da rede: a transferência do arquivo e o compartilhamento do arquivo no dispositivo original. Na prática, os bytes que compõem o arquivo são transferidos do dispositivo de origem para o de destino nos dois métodos. Mas no segundo método, as informações alteradas voltam ao dispositivo original por meio da rede, mantendo as informações centralizadas.

Além da transmissão do conteúdo de um arquivo, é importante também a verificação de informações auxiliares (propriedades) que estão associadas ao arquivo e como elas serão tratadas no dispositivo de destino.

### 8.1.1 Propriedades dos arquivos

As propriedades dos arquivos indicam quando o arquivo foi criado, o tamanho total do arquivo, que tipo de restrições existem para a manipulação do conteúdo do arquivo e, num ambiente com diferentes usuários, qual usuário ou grupo pode ter acesso ao arquivo.

Um mesmo sistema operacional pode utilizar diferentes sistemas de arquivos e cada sistema destes trata estas características de uma forma específica.

fica. Ao transferir o arquivo, deve haver uma alternativa para transferir algumas destas propriedades ou então assumir no sistema de arquivos do destinatário novas características.

## 8.2 Transferência de arquivos

A transferência de arquivo consiste em transferir o arquivo de um dispositivo para outro sem que o arquivo no dispositivo de destino mantenha algum vínculo com o arquivo original. Para todos os efeitos, o arquivo transferido para o dispositivo de destino é um novo arquivo, que pode ou não, manter as características de data de criação, nome ou última alteração do arquivo original.

Do mesmo modo que a cópia de um arquivo de uma estrutura de pastas para outra na mesma estrutura ou para meios magnéticos temporários, a transferência de um arquivo na rede depende do sistema de destino para o controle de espaço e localização.

### 8.2.1 Protocolo FTP

Dentro da arquitetura TCP/IP, um dos protocolos pioneiros na transferência de arquivos foi o protocolo FTP (*File Transfer Protocol*) que foi proposto em 1971, no início da ARPANET.

Este protocolo atua no formato cliente-servidor e cria um canal de controle em uma porta TCP 21 no servidor, e neste canal processa os comandos que são usados para selecionar o local, o arquivo e executar a transferência.

Para a transferência dos dados do arquivo, é utilizado um outro canal através da porta TCP 20 no servidor.

Para a utilização deste tipo de transferência de arquivos é necessário instalar no servidor um serviço que implementa o protocolo. Neste capítulo estaremos usando como referência o servidor “VsFTPD” que pode ser instalado no GNU/Linux Mint através do comando “`apt-get install vsftpd`”.

Para a parte cliente temos o comando “`ftp`”. Ele possui vários subcomandos para as operações com arquivos e pastas (vistos no quadro 8.2). Este comando já vem pré-instalado na maioria das distribuições GNU/Linux

(no Windows podem ser utilizados os aplicativos FileZilla ou WinSCP como cliente).

Quadro 8.1 – Conectando em um servidor FTP (VsFTPd) no GNU/Linux

```
1. # ftp 192.168.0.12
2. Connected to 192.168.0.12.
3. 220 (vsFTPD 3.0.2)
4. Name (192.168.0.12:root): usuario
5. 331 Please specify the password.
6. Password:
7. 230 Login successful.
8. Remote system type is UNIX.
9. Using binary mode to transfer files.
10. ftp>
```

Fonte: Elaborado pelo o autor

No exemplo do quadro 8.1, temos a conexão ao servidor 192.168.0.12 na linha 1 e nas linhas 4 e 6 são solicitados o nome de usuário e a senha, respectivamente, para o acesso ao sistema de arquivos do servidor.

Na linha 10, após ter aceito o usuário e senha, o programa aguarda que seja indicado um comando para ser executado no servidor. Uma lista dos principais comandos encontra-se no quadro 8.2.

A partir do indicador da linha 10, basta digitar os comandos para a navegação no sistema de arquivos do servidor.

Quadro 8.2 – Principais comandos do protocolo FTP

Comando	Descrição
user	Inicia a identificação do usuário. Pode ser enviado também a senha ao lado do nome do usuário.

Comando	Descrição
cd	Permite navegar pelas pastas do servidor. Mesmo conceito do CD do Linux.
quit	Termina a conexão com o servidor FTP.
passive	Indica que a conexão será feita a partir do cliente para passar pelo firewall
get	Transfere o arquivo indicado para a máquina local
put	Envia o arquivo para o servidor. No vsftpd é necessário habilitar no arquivo de configuração ( <i>/etc/vsftpd.conf</i> ) a permissão para salvar o arquivo no servidor ( <i>write_enable=YES</i> ).
pwd	Mostra em qual pasta no servidor se encontra o usuário conectado.
ls / dir	Mostra os arquivos e pastas do local atual no servidor.

Fonte: Elaborado pelo autor.

E para transferir um arquivo para o servidor, usa-se o comando PUT. Já para recuperar um arquivo temos o comando GET. No quadro 8.3 pode ser visto um exemplo de envio de arquivo para o servidor.

Quadro 8.3 – Navegando e copiando arquivos em um servidor FTP (VsFTPD) no GNU/Linux

```

1.
2. ftp> ls
3. 200 PORT command successful. Consider using PASV.
4. 150 Here comes the directory listing.
5. drwxr-Xr-X 3 1000 1000 4096 Apr 30 2016 Documentos
6. drwxr-Xr-X 14 1000 1000 4096 Jul 02 13:27 Downloads
7. drwxr-Xr-X 2 1000 1000 4096 Feb 27 2016 Imagens
8. drwxr-Xr-X 2 1000 1000 4096 Feb 11 2016 Modelos
9. drwxr-Xr-X 10 1000 1000 4096 Nov 09 22:43 firefox
10. 226 Directory send OK.
11. ftp> put livro.pdf
12. local: livro.pdf remote: livro.pdf
13. 200 PORT command successful. Consider using PASV.
14. 150 Ok to send data.
15. 226 Transfer complete.
16. 1419542 bytes sent in 1.44 secs (964.2 kB/s)
17. ftp>

```

Fonte: Elaborado pelo autor.

Após a conexão (ver quadro 8.1), o sistema fica aguardando o uso dos comandos citados no quadro 8.2. Na linha 2 é mostrado o conteúdo da pasta onde foi iniciada a conexão e na linha 11 é usado o comando PUT para enviar o arquivo “livro.pdf” para o servidor. Na linha 12 é mostrado que foi utilizado o mesmo nome no destino e na linha 16 é informado qual a quantidade de bytes enviados e o tempo que levou esta transmissão.

O FTP em seu modo padrão transmite a senha do usuário sem criptografia. Este tipo de situação permite que algum ponto da rede, onde possa executar um “tcpdump” ou “wireshark” (vistos nos capítulos anteriores) pode capturar esta informação, permitindo que, posteriormente, esta conta de FTP possa ser utilizada. Nestes casos, pode ser utilizado um túnel de criptografia para passar as informações e garantir que não sejam visualizadas durante a transferência.

Outro ponto de cuidado com servidores FTP é a possibilidade do usuário conectado poder navegar dentro da estrutura de pastas do servidor. Este tipo de situação permitiria que durante a conexão via FTP, o usuário pudesse retirar arquivos que não estivessem em sua pasta de trabalho.

## Saiba mais

Em situações onde não se deseja que o usuário possa navegar em outras pastas do servidor é possível habilitar uma configuração para que essa navegação seja restrita a pasta de trabalho padrão. O uso do comando “chroot” faz com que a área de trabalho do usuário seja convertida para uma estrutura raiz. O uso do “chroot” em conjunto com o servidor FTP auxilia a melhorar o uso do serviço de cópia e envio de arquivos.

Apesar da idade, o uso do serviço FTP ainda é uma das formas de envio de arquivos para servidores remotos, que podem estar localizados em provedores de serviço. Um exemplo são os serviços de hospedagem de sites que permitem a atualização dos arquivos via FTP.

O uso de interface gráfica para a manipulação dos arquivos de maneira remota pode ser utilizado e inclusive, alguns gerenciadores gráficos de arqui-

vos locais em Linux implementam o protocolo FTP sendo possível o uso destes gerenciadores com cliente FTP.

### 8.2.2 Protocolo HTTP (Hypertext Transport Protocol)

Como foi verificado nos dois últimos capítulos, o HTTP tem sido utilizado também para a transferência de documentos não relacionados a hipertexto. Com isso, o servidor HTTP tem substituído em muitos cenários a necessidade de um servidor FTP para a distribuição simples de arquivos.

## 8.3 Compartilhamento de arquivos

Na década de 1980, era alto o custo dos meios de armazenamento e de impressão. A proliferação das redes locais nas empresas nesta época veio com esta intenção de interligar as máquinas para compartilhar recursos.

Enquanto que seguia o desenvolvimento da ARPANET, algumas empresas de tecnologia investiram no desenvolvimento de tecnologias de rede específicas para o uso local. O XNS foi uma iniciativa da empresa Xerox que tornou pública sua especificação. Esta proposta influenciou diversas outras redes de empresas como 3COM, Netware e Banyan Vines (CISCO, 2016). A rede e os serviços da Netware foram sinônimos de rede local durante alguns anos.

Estas redes locais criavam métodos para a utilização dos recursos da máquina (por exemplo, o disco local) remotamente por outros dispositivos da rede. Nestes casos, um disco magnético com maior capacidade poderia centralizar documentos que poderiam ser acessados por outros dispositivos, ou uma impressora poderia ser usada por um equipamento em outro andar de um prédio sem a necessidade de levar o arquivo a ser impresso para a máquina que estava conectada na impressora.

Algumas máquinas com maior capacidade de processamento eram escolhidas para concentrar os recursos principais (disco magnético, leitor de CDROM, unidades de *backup*) e, além do uso normal executando aplicativos locais, eram acessadas pelos demais participantes da rede. A medida que aumentava o uso dos equipamentos remotos, estas máquinas começaram a ficar dedicadas para o compartilhamento dos recursos.

Esta estratégia de máquinas de maior capacidade, com a centralização de recursos e serviços de compartilhamento ainda é utilizada no projeto das redes locais das empresas, onde um equipamento servidor, com um sistema operacional preparado para o compartilhamento de diferentes tipos de recursos, é utilizado como base para a rede local. Os sistemas mais utilizados para esta função são os da família Windows Server e GNU/Linux.

### 8.3.1 Protocolo SMB (Server Message Block)

Como as redes corporativas utilizam principalmente o sistema operacional Windows nos *desktops*, as redes locais popularizaram os protocolos de compartilhamento usados neste sistema.

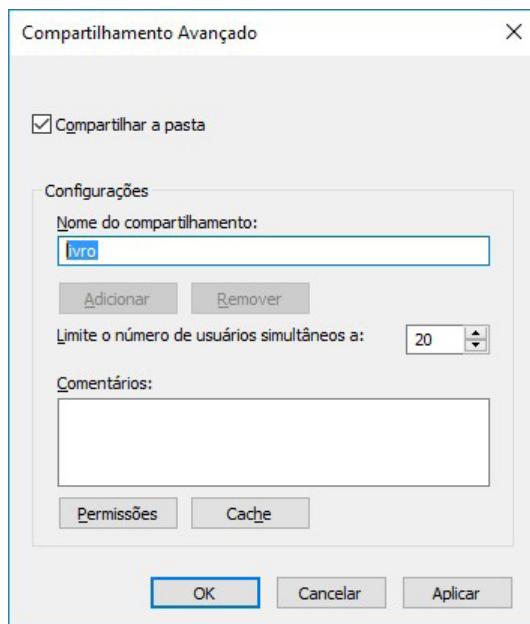
Estes protocolos usados nas redes Windows atuais, foram baseados nos serviços desenvolvidos via uma API (*Application Programming Interface*) de referência usada para a comunicação dos sistemas IBM PC em redes locais chamada de NetBIOS desenvolvidos pela IBM. A partir desta API foi criado pela IBM o SMB (server Message Block), que foi evoluído principalmente pelas empresas 3Com, Microsoft, Intel e adotado pela Microsoft como base para sua proposta de serviços de compartilhamento de arquivos e impressão em rede local com Windows.

O termo CIFS (*Common Internet File System*) é empregado para um dialeto do SMB original, utilizado nos sistemas Windows NT e Windows 98. Com o Windows 2000 inicia a utilização da especificação SMB 1.0 e a partir do Windows Vista, a especificação SMB2 (e suas evoluções) foram disseminadas tanto para as versões de sistema *desktop* quanto para servidores (Windows Server).

Estes protocolos utilizam as portas 137, 138 e 139 durante a comunicação conforme a versão utilizada e o tipo de mensagem, e nos casos em que é feita a utilização direta do TCP como transporte é usada a porta 445.

Na figura 8.1, pode ser verificada a tela de criação de um compartilhamento de uma pasta num sistema Windows 10, que será visualizada na rede como um compartilhamento chamado “livro”. Não é necessário que o nome do compartilhamento seja igual ao nome da pasta que foi selecionada.

Figura 8.1 – Criação de compartilhamento em Windows 10

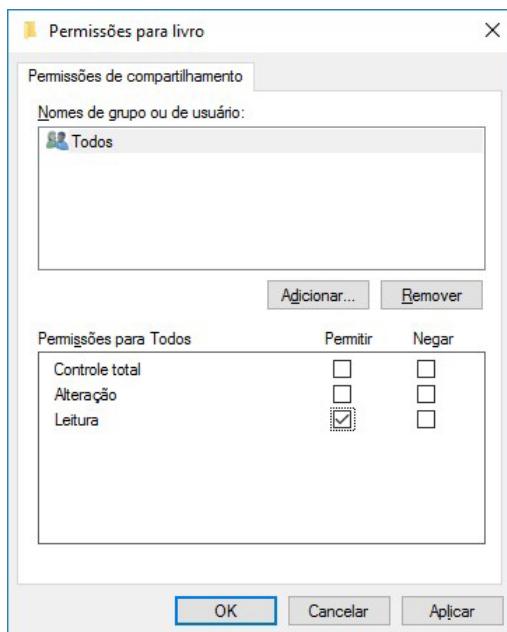


Após a indicação do nome do compartilhamento, no botão “permissões” pode ser verificado quais usuários poderão ter acesso a este compartilhamento e que tipo de permissão podem ter: apenas acesso de leitura nos arquivos disponíveis no compartilhamento, acesso de alteração e controle total nos arquivos do compartilhamento.

Estas características podem ser associadas a um grupo de usuários ou a usuários específicos. Podem ser criadas combinações entre opções liberadas ou negadas para cada usuário se necessário. Na figura 8.2, tem-se a janela de configuração de permissão para o compartilhamento “livro” feito anteriormente, com a identificação do grupo de todos os usuários do sistema, que poderão apenas fazer a leitura dos arquivos que estiverem neste compartilhamento.

Esta sequência permite que um dispositivo remoto possa acessar o compartilhamento “livro” e abrir ou copiar o arquivo para o ambiente local. Mas não poderá fazer alterações, apagar ou criar novos arquivos e pastas dentro deste compartilhamento.

Figura 8.2 – Definição de permissão de acesso a um compartilhamento



Ao utilizar a linha de comando no sistema Windows, estes compartilhamentos podem ser verificados com o uso do comando “net”. No quadro 8.4, pode ser verificada a lista de compartilhamentos, incluindo o compartilhamento “livro” (linha 10) feito nos exemplos acima, ao executar o comando “net share” (linha 1).

Quadro 8.4 – Listando compartilhamentos via linha de comando do Windows 10

Nome	Recurso	Observação
5. C\$	C:\	Recurso compartilhado padrão
6. D\$	D:\	Recurso compartilhado padrão
7. IPC\$		IPC remoto
8. print\$	C:\WINDOWS\system32\spool\drivers	Drivers de impressora
9. ADMIN\$	C:\WINDOWS	Administração remota
10. livro	C:\Users\teste\Documents\livro\smb	
11. Users	C:\Users	
12.		Comando concluído com êxito.

Fonte: Elaborado pelo autor.

Estes compartilhamentos podem ficar associados a letras dentro do sistema Windows. Neste sistema é padrão fixar os recursos externos a uma letra, por exemplo uma unidade de DVD ou um *pendrive* pode ser acessado clicando na letra correspondente. No caso de compartilhamentos de rede, cada vez que é selecionada uma letra associada a este compartilhamento em uma janela do gerenciador de arquivos, são mostrados os itens disponíveis. Caso esta associação de letras seja gravada no sistema, quando o usuário se identifica novamente, o compartilhamento é associado de forma automática.

### 8.3.2 Sistema Samba

Desenvolvido no início da década de 1992 como um projeto pessoal do programador Andrew Tridgell, o sistema Samba desenvolvido para Linux permitiu acessar recursos compartilhados com o protocolo SMB entre Windows e Linux (SAMBA, 2016). Este sistema é uma alternativa *opensource* de implementação do protocolo SMB/CIFS e se tornou uma opção robusta para o uso em empresas e demais cenários onde os dois sistemas precisam operar simultaneamente.

No quadro 8.5 foi utilizado o comando *smbclient* num sistema Linux que acessa um dispositivo na rede para listar as opções de compartilhamentos de um dispositivo remoto com sistema Windows 10. Para este objetivo foi utilizada a opção “*-L*” para listar os compartilhamentos e a opção “*-U*” para indicar o usuário “*yy*” que foi liberado para o acesso a este compartilhamento. Dentre as opções padrão listadas pelo sistema está o compartilhamento “*livro*” (linha 13) que foi criado no exemplo anterior. Na linha 5 são mostradas as informações sobre o sistema do dispositivo remoto (Windows 10) que está compartilhando uma pasta para o usuário “*yy*”. Compare o resultado do quadro 8.5 com o quadro 8.4, onde foi utilizado o comando “*net share*” na linha de comando do Windows 10.

Quadro 8.5 – Listando os compartilhamentos de Windows 10 em GNU/Linux

```
1. # smbclient -L 192.168.0.19 -U yy
2.
3. Enter yy's password:
4.
5. Domain=[TESTE] OS=[Windows 10 Pro 14393] Server=[Windows 10 Pro 6.3]
6.
7.   Sharename      Type      Comment
8.   -----
9.   ADMIN$        Disk      Administração remota
10.  C$            Disk      Recurso compartilhado padrão
11.  D$            Disk      Recurso compartilhado padrão
12.  IPC$          IPC       IPC remoto
13.  livro          Disk      Drivers de impressora
14.  print$        Disk      Drivers de impressora
```

Fonte: Elaborado pelo autor.

No quadro 8.6 foi novamente feito o acesso ao mesmo dispositivo, desta vez indicando o compartilhamento que deve ser acessado no dispositivo “//192.168.0.19/livro”. Novamente foi utilizado o usuário “yy” que tem a permissão de acesso ao compartilhamento. Ao finalizar o processo de conexão o sistema fica aguardando a digitação de algum comando. No exemplo, foi utilizado o comando “ls” (linha 7) para listar os arquivos que estão dentro do compartilhamento. O retorno do comando é uma listagem de todos os arquivos, onde na linha 10 está o arquivo “telas.docx” que foi disponibilizado para este exemplo.

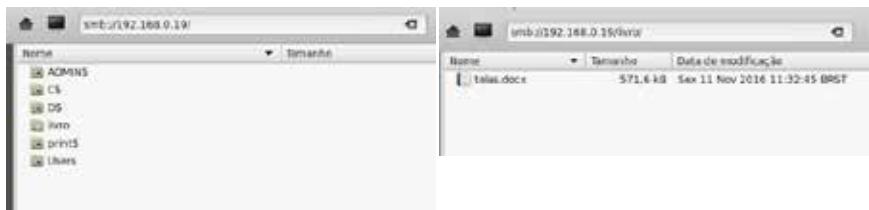
Quadro 8.6 – Conectando num compartilhamento privado e listando o seu conteúdo

```
1. # smbclient -U yy //192.168.0.19/livro
2.
3. Enter yy's password:
4.
5. Domain=[TESTE] OS=[Windows 10 Pro 14393] Server=[Windows 10 Pro 6.3]
6.
7. smb: \> ls
8. .
9. ..
10. telas.docx
11.
12. 94579199 blocks of size 4096. 46999527 blocks available
```

Fonte: Elaborado pelo autor.

Os compartilhamentos feitos no sistema Windows 10 também podem ser verificados utilizando uma interface gráfica, que permite mostrar as mesmas informações verificadas no retorno dos comandos do Samba executados nos exemplos anteriores. Mas as informações são formatadas para dar a mesma aparência da visualização de arquivos locais. Na figura 8.3, as duas telas mostram no gerenciador de arquivos do GNU/Linux Mint, os compartilhamentos disponíveis no Windows 10 (acessado pelo IP 192.168.0.19 na barra superior) e após clicar no compartilhamento “livro”, é listado o arquivo “telas.docx” que está salvo remotamente.

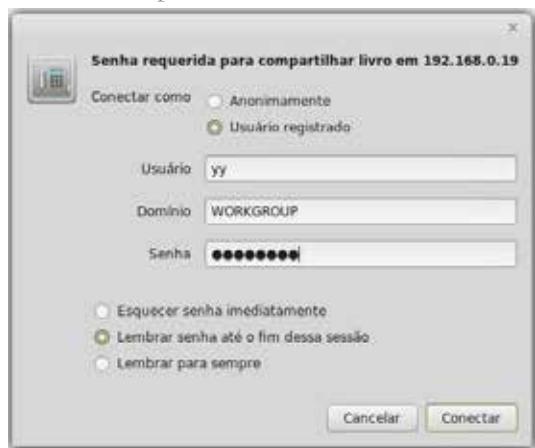
Figura 8.3 – Conectando em um compartilhamento Windows 10 usando o gerenciador de arquivos do GNU/Linux Mint



Como no Windows 10 foi configurado um compartilhamento privado, para acesso ao usuário “yy” apenas, ao fazer a conexão com o sistema Windows remoto é solicitado o usuário e senha para permitir o acesso. Na figura 8.4 é mostrada a tela de solicitação de acesso ao compartilhamento “livro” quando é inserido o endereço IP do dispositivo que tem o sistema Windows 10 e o compartilhamento indicado neste exemplo.

Após a digitação do usuário “yy” e a senha, pode ser indicado por quanto tempo o programa deve guardar a informação do acesso. Ao clicar no botão conectar e se a senha

Figura 8.4 – Digitação de usuário e senha para acesso ao compartilhamento “livro”



estiver correta, será mostrada a tela com os arquivos que estão disponíveis neste compartilhamento. No caso destes exemplos utilizados, o arquivo “telas.docx” é listado com suas propriedades sendo visíveis no dispositivo remoto. Ao ter acesso ao compartilhamento, conforme o tipo de permissão liberado para o usuário, é possível criar arquivos ou pastas, apagar, copiar ou listar o conteúdo do compartilhamento.

Enquanto que o protocolo SMB se estabeleceu nas redes locais de microcomputadores com o uso de sistemas operacionais monousuários, no uso de equipamentos de maior capacidade que utilizavam sistemas baseados em Unix, o padrão para compartilhamento de arquivos foi o protocolo NFS (*Network File System*).

### 8.3.3 Protocolo NFS (Network File System)

Em 1984, a empresa Sun Microsystems criou um protocolo de aplicação para o compartilhamento de um sistema de arquivos completo. Este protocolo (NFS), vem com o objetivo de ser independente de sistema operacional e cria um modelo genérico de sistema de arquivos para ser utilizado pelos diferentes sistemas.

O NFS é um protocolo cliente-servidor e foi padronizado para a Internet na sua segunda versão em 1989 (RFC1094, 2016). O servidor compartilha um sistema de arquivos e os clientes podem acessar este sistema simultaneamente usando a porta TCP 2049.

#### Saiba mais

Novas versões do NFS estão possibilitando o uso de compartilhamentos em **paralelo** (com vários dispositivos de armazenagem), e vão auxiliar no acesso a grandes volumes de dados. O crescimento da capacidade da rede Ethernet (10G, 100G), a disponibilidade de HD de estado sólido (SSD) e o grande volume de dados impulsionam as melhorias nos protocolos de compartilhamento.

Diferente da proposta dos sistemas Windows, nos sistemas baseados em Unix, como é o caso do GNU/Linux Mint usado como referência, quando um

recurso externo é fixado no sistema não é associado a uma letra. O padrão Unix é anexar esta nova estrutura de arquivos na estrutura existente, fazendo com que ela mantenha a continuidade da estrutura dos arquivos. Por exemplo, caso o sistema tenha uma pasta “jogos” que esteja na estrutura “/local/usuario/jogos” e for anexada uma estrutura de um servidor remoto que contenha o compartilhamento “jogo-de-xadrez”, ele ficará anexado à estrutura como se fosse uma pasta local “/local/usuario/jogos/jogo-de-xadrez”. Os comandos normais de navegação nas pastas (ou aplicativos gráficos de gerenciamento de arquivos) podem navegar para dentro do compartilhamento e copiar ou criar novos arquivos conforme a permissão utilizada no compartilhamento da estrutura.

Para iniciar um compartilhamento usando o NFS é preciso indicar no arquivo “exports” na pasta “/etc”, qual a pasta será compartilhada e pode ser indicado qual rede ou IP pode acessar o compartilhamento. No quadro 8.7 pode ser verificada uma linha do arquivo “exports” em que é identificada uma pasta que pode ser acessada por qualquer equipamento da rede 192.168.0, mas com permissão apenas de leitura “ro” (*read only*).

**Quadro 8.7 – Configurando um compartilhamento no arquivo “/etc(exports” no Linux**

```
/var/livro/ 192.168.0.0/24(ro,sync,no_subtree_check)
```

Fonte: Elaborado pelo autor.

Sempre que for alterado o arquivo “exports” é necessário indicar ao serviço NFS a mudança. Isto pode ser feito reiniciando o serviço do NFS, como verificado no quadro 8.8, ou pode ser utilizado o comando “exportfs -r”.

**Quadro 8.8 – Reiniciando o serviço do NFS**

```
1. # service nfs-kernel-server restart
2. * Stopping NFS kernel daemon [ OK ]
3. * Unexporting directories for NFS kernel daemon... [ OK ]
4. * Exporting directories for NFS kernel daemon... [ OK ]
5. * Starting NFS kernel daemon [ OK ]
```

Fonte: Elaborado pelo autor.

Para verificar quais as opções de compartilhamento, o comando “showmount” pode ser usado. No quadro 8.9 pode ser verificado o resultado do comando em duas opções. Na primeira o comando é executado no lado do servidor para identificar quais os compartilhamentos estão ativos. Na segunda opção (lado direito) está o retorno do comando executado em um dispositivo remoto que fez a consulta ao servidor (192.168.0.13).

Quadro 8.9 – Conferindo o compartilhamento NFS

Consulta local	Consulta remota
1. # showmount -e	1. # showmount -e 192.168.0.13
2. Export list for YY2:	2. Export list for 192.168.0.13:
3. /var/livro 192.168.0.0/24	3. /var/livro 192.168.0.0/24

Fonte: Elaborado pelo autor.

Para o cliente anexar este compartilhamento no dispositivo local, ele executa o comando “mount” com a opção “-t nfs” que indica que será anexado uma estrutura de arquivos numa pasta local. Neste caso, a estrutura de arquivos é de um dispositivo remoto. No quadro 8.10, é mostrada a sintaxe de um compartilhamento sendo anexado.

Quadro 8.10 – Anexando o sistema remoto na pasta local

```
1. # mkdir teste
2. # mount -t nfs 192.168.0.13:/var/livro teste
3.
4. # cd teste
5. # ls
6. capitulo8.txt HarryPotter.txt tcp.png
7.
8. # mkdir pasta-temporaria
9. mkdir: cannot create directory 'pasta-temporaria': Read-only file
   system
```

Fonte: Elaborado pelo autor.

No quadro anterior, na linha 2 pode ser verificado o comando “mount” que está solicitando acesso ao servidor 192.168.0.13 para a pasta compartilhada “/var/livro” que será anexada à pasta local “teste”.

Ao terminar a montagem na pasta local, o compartilhamento fica anexado como uma pasta da estrutura do sistema de arquivos local. Utilizando o comando

“*cd teste*” na linha de comando do Linux (linha 4) entra-se na pasta teste e com o comando “*ls*” são listados os arquivos disponíveis pelo servidor. Não se percebe nenhuma diferença comparado ao comportamento de uma pasta local.

Como o compartilhamento no lado servidor foi criado com a permissão “ro” de leitura, ao tentar criar uma pasta (linha 8) o sistema emite uma mensagem de erro indicando que não pode criar a ‘pasta-temporaria’.

Caso seja necessário criar ou alterar algo no compartilhamento pelo dispositivo remoto, pode ser alterado no servidor, no arquivo “exports” a opção para “rw” (read & write) indicando que pode ser feita a leitura e escrita no compartilhamento.

Como o arquivo foi alterado, usa-se o comando “*exportfs -r*” e deve ser refeito o compartilhamento no lado cliente novamente. Para isso, primeiro desmonta-se o compartilhamento “*umount teste*” e executa novamente o compartilhamento como no quadro 8.10.

No quadro 8.11, foi utilizado o comando “*mount*” com a opção “-t nfs” para verificar a situação da nova conexão via NFS.

Quadro 8.11 – Verificando o compartilhamento ativo

```
1. # mount -t nfs  
2.  
3. 192.168.0.13:/var/livro on /root/teste type nfs (rw,  
vers=4,addr=192.168.0.13,clientaddr=192.168.0.12)
```

Fonte: Elaborado pelo autor.

Neste caso, pode ser verificado na linha 3, que o compartilhamento foi feito com “rw” entre o servidor 192.168.0.13 e o cliente 192.168.0.12, na versão de NFS 4. A partir deste momento é possível criar pastas e arquivos no compartilhamento.

Para que um compartilhamento seja anexado automaticamente sempre que é iniciado o sistema, deve ser acrescentado uma linha no arquivo “*fstab*” na pasta “*/etc/*”. Neste arquivo deve ser inserida uma linha contendo a pasta remota a ser anexada e as opções do compartilhamento.

Quadro 8.12 – Criando uma montagem automática do compartilhamento NFS

```
192.168.0.13:/var/livros /root/teste nfs auto 0 0
```

Fonte: Elaborado pelo autor.

No quadro 8.12, a linha do “fstab” que vai permitir que o compartilhamento seja criado automaticamente sempre que o sistema inicie.

## 8.4 Dispositivos de armazenamento em rede

Com o volume de dados em crescimento e a quantidade de dispositivos em uma rede aumentando, o desempenho no acesso aos arquivos é uma importante questão no planejamento de uma rede. Também algumas situações de segurança e continuidade de negócio impõem ao administrador de redes que se preocupe em como estes dados estejam sendo cuidados para que no caso de uma pane, seja possível acessar novamente as informações armazenadas.

A centralização do armazenamento de dados em um dispositivo centralizado, e fazendo com que todos os demais dispositivos guardem as informações nesta central de arquivos, cria um meio controlado para poder implementar tecnologias para a cópia de segurança destas informações.

Um servidor tem sido utilizado para a centralização destes arquivos em várias redes locais. Mas também são utilizados dispositivos específicos de armazenamento, os chamados “*storages*”.

Alguns destes dispositivos específicos de armazenamento já vêm com opções de compartilhamento incorporada em seus sistemas. Também possuem hardware dedicado para o controle dos discos e sistemas de arquivos. Conforme as características específicas de proteção estes dispositivos podem crescer também em relação ao investimento necessário.

No início, quando as redes tinham baixa capacidade e desempenho, os dispositivos de armazenamento dedicados precisavam de uma rede específica para a transferência de dados entre eles e os servidores. Esta rede especial chamada de SAN (*Storage Area Network*) é cara e tem *hardware* e tecnologias especiais, sendo a mais conhecida a *fibre channel*. Atualmente, também pode ser utilizada a rede normal dos computadores para a transferência de dados

dos dispositivos de armazenamento. Este tipo de estrutura que usa a rede local é chamado de NAS (*Network Attached Storage*).

### 8.4.1 NAS (Network Attached Storage)

Os equipamentos NAS ficam conectados diretamente na rede local, podendo ser acessados por todos os demais equipamentos e proveem mecanismos de controle dos discos. Uma diferença básica entre os dispositivos que fazem parte de uma estrutura NAS e os que fazem parte de uma estrutura SAN é que os primeiros trabalham com arquivos e os segundos trabalham em mais baixo nível, com blocos de bytes. Por este motivo, os equipamentos NAS implementam os protocolos na camada de aplicação vistos neste capítulo, como SMB/CIFS, NFS e FTP em sua interface de acesso.

Com o crescimento no desempenho e capacidade das redes Ethernet com a disseminação de redes 10G, a utilização deste tipo de tecnologia (SoE – *Storage over Ethernet*) tem demonstrado uma tendência para a infraestrutura de armazenamento em *data centers*.

Enquanto que um hardware específico pode trazer benefícios com relação a desempenho e controle de discos, um serviço NAS pode ser criado utilizando um computador e um *software* de gerenciamento de arquivos. Podem ser utilizados NAS proprietários com características e robustez variadas e também é possível utilizar um sistema NAS *opensource*. Um exemplo que pode ser utilizado gratuitamente é o sistema FreeNAS, que proporciona acesso a um

Figura 8.5 – Tela de exemplo de uma instalação do FreeNAS (gerenciamento de volumes)



sistema de arquivos centralizado por meio de diferentes protocolos de acesso (NFS, SMB, FTP, iSCSI).

Na figura 8.5 pode ser verificado um exemplo da interface do sistema FreeNAS. Neste sistema *opensource*, um repositório pode ser montado utilizando vários discos em um computador e compartilhando estes volumes com protocolos de compartilhamento e transferência de arquivos, sendo que o sistema é configurado para que este ambiente fique dedicado a esta tarefa. Como um equipamento montado para este objetivo, o FreeNAS toma conta do equipamento e se instala como um sistema operacional dedicado, não sendo possível o uso para outras tarefas de um sistema operacional geral. Desta forma, os recursos de processamento e acesso a disco ficam dedicados ao compartilhamento via rede.

## 8.5 Armazenamento em Nuvem

Com a evolução dos provedores de serviço na Internet, vem crescendo o uso de infraestrutura como um serviço (IaaS – *Infrastructure as a Service*), onde os servidores e equipamentos de armazenagem (*Storages*) ficam instalados fora da empresa, e seus recursos (processamento e disco) são acessados via Internet.

### Saiba mais

Para a pessoa física também podem ser utilizados serviços na nuvem para armazenamento de dados e cópia de segurança de arquivos. Empresas como Google, Dropbox, Microsoft e outras, já permitem a criação de pastas e repositórios para sincronização de arquivos entre dispositivos como *smartphones*, *tablets* e *notebooks* com servidores na Internet.

Uma das características do uso de armazenamento de dados em nuvem (*Cloud*), é a possibilidade de solicitar o tipo de recurso (e o aumento ou diminuição da capacidade contratada) **por demanda**. Assim a empresa pode expandir sua capacidade de processamento e armazenagem sem a necessidade de preparar o ambiente antecipadamente (rede, energia, refrigeração, backup). O investimento também pode ser gradual, pagando apenas o que consome a mais e, principalmente, se a necessidade deixa de existir, não fica com recurso ocioso.

Um dos principais fornecedores deste tipo de recurso em nuvem é a Amazon. Esta empresa possui diversos tipos de recursos que podem ser contratados por demanda, sendo um dos serviços o S3 (*Simple Storage Service*): serviço de armazenamento em nuvem.

### 8.5.1 Amazon S3

O S3 pode ser contratado para a centralização de arquivos na nuvem da Amazon, mas além da simples armazenagem, também é possível configurar algumas propriedades de versionamento dos dados e de validade da informação. Estes dados podem ter acesso restrito para a empresa ou podem ser acessados publicamente. Neste último caso, o próprio S3 pode ser utilizado como um servidor WEB (site) de páginas estáticas (ver capítulo 7). Na figura 8.6, pode ser vista a tela do sistema S3. Nesta tela temos na parte superior um botão “Create Bucket” que é a maneira de criar uma estrutura para o armazenamento de arquivos. A partir desta estrutura, podem ser criadas pastas e copiados arquivos.

Figura 8.6 – Tela de exemplo de um armazenamento no S3

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with a 'Create Bucket' button and an 'Actions' dropdown. Below that is a list titled 'All Buckets (6)' containing the following buckets:

- saopaulo
- livro** (highlighted in blue)
- santossp
- riodejai
- desenvol
- vitoriaf

On the right, the main panel displays the properties for the selected bucket 'livro'. It includes the following information:

- Bucket:** livro
- Region:** Oregon
- Creation Date:** Tue Nov 15 06:31:19 GMT-100 2016
- Owner:** Itaíres

Below this, there's a sidebar with several expandable sections:

- Permissions
- Static Website Hosting
- Logging
- Events
- Versioning
- Lifecycle
- Cross-Region Replication
- Tags
- Requester Pays

Foi criado como exemplo um *bucket* chamado “livro” e ao ser selecionado, é mostrado ao lado as propriedades desta estrutura. Pode ser criada

uma regra que ao final de dezembro de 2016, todos os arquivos são apagados. Ou que todo arquivo com mais de 6 meses pode ser apagado da estrutura.

## 8.6 Utilitários

Os utilitários a seguir podem ser utilizados para transferência de arquivos entre diferentes equipamentos e dispositivos de armazenagem, utilizando a linha de comando. Usando os serviços de agendamento do Linux (cron) e do Windows (Agendador de tarefas), podem ser automatizadas as transferências dos dados.

### 8.6.1 AWS

Este utilitário disponibilizado pela Amazon pode ser utilizado tanto no Linux quanto no Windows e permite que sejam feitas operações entre o sistema cliente e diversos serviços da Amazon via linha de comando. Um dos serviços é o S3 e com este utilitário, podem ser feitas copias de arquivos de e para o S3.

Quadro 8.13 – Listando conteúdo do *bucket* S3

```
# aws s3 ls s3://livro
2015-10-09 13:58:00      1095 bkp.bat
2016-10-25 17:03:28    12878506 myprojects.7z
2016-10-26 11:21:36  138939646 system_p12_migracao.7z
```

Fonte: Elaborado pelo autor.

No quadro 8.13, foi utilizado o comando *aws* com a opção “*s3*”, indicando que as opções referem-se ao serviço de armazenagem, a opção “*ls*” apontando que deseja listar o conteúdo e “*s3://livro*” indicando qual *bucket* é para ser acessado.

Podem também ser utilizadas as opções: *cp* utilizado para copiar arquivos, *mv* usado para mover arquivos, *rm* remove (apaga) arquivos e *sync* que permite deixar sincronizada uma pasta local ou remota. No quadro 8.14, um

exemplo de cópia de um arquivo do *bucket* “livro” para a pasta local (o ponto ao final da primeira linha indica a pasta local).

Quadro 8.14 – Copiando conteúdo do *bucket* S3

```
# aws s3 cp s3://livro/bkp.bat .
download: s3://livro/bkp.bat to ./bkp.bat
```

Fonte: Elaborado pelo autor.

A opção *sync* utiliza a data de modificação do arquivo para avaliar se é mais novo ou mais antigo que o repositório a ser sincronizado. Após a execução do comando as duas pastas ficam com os mesmos conteúdos, sendo uma opção para criar cópias de segurança de estruturas inteiras de pastas de um servidor local para a nuvem.

### 8.6.2 Rsync (Remote Sync)

O comando *rsync* é utilizado como opção para a cópia de arquivos e também para sincronizar pastas, mantendo os mesmos arquivos entre elas. Pode ser utilizado para sincronizar pastas entre dois dispositivos remotos e também pode ser utilizado para sincronia de pastas locais (no mesmo disco ou em discos externos).

Quadro 8.15 – Copiando conteúdo entre pastas usando *rsync*

```
1. # rsync -a livro/ livro-bkp/
2.
3. YY2 var # ls -l livro*
4. livro:
5. -rw-r--r-- 1 root root 1095 Out 9 2015 bkp.bat
6. -rw-r--r-- 1 root root 160 Nov 14 18:53 capitulo8.txt
7. -rw-r--r-- 1 root root 332 Nov 14 18:53 HarryPotter.txt
8. -rw-r--r-- 1 root root 492 Nov 14 18:54 tcp.png
9. -rw-r--r-- 1 nobody nogroup 102 Nov 14 20:10 teste.txt
10.
11. livro-bkp:
12. -rw-r--r-- 1 root root 1095 Out 9 2015 bkp.bat
13. -rw-r--r-- 1 root root 160 Nov 14 18:53 capitulo8.txt
14. -rw-r--r-- 1 root root 332 Nov 14 18:53 HarryPotter.txt
15. -rw-r--r-- 1 root root 492 Nov 14 18:54 tcp.png
16. -rw-r--r-- 1 nobody nogroup 102 Nov 14 20:10 teste.txt
```

Fonte: Elaborado pelo autor.

No quadro 8.15, o comando *rsync* (linha 1) foi executado com a opção “*-a*” indicando que é para manter as propriedades dos arquivos de origem na cópia, e fez a cópia dos arquivos da pasta “*livro*” para a pasta “*livro-bkp*”. Nas linhas 4 a 16 podem ser vistos os arquivos que foram copiados com as mesmas informações de data e hora.

Ao ser utilizado para sincronizar as pastas, o *rsync* faz uma verificação com relação a tamanho e data/hora de modificação do arquivo para decidir se efetua a cópia ou não entre as pastas. No quadro 8.16, pode ser verificado que na pasta original “*livro/*” o arquivo “*capitulo8.txt*” foi modificado e está com o tamanho e data/hora mais recente (comparar com quadro anterior 8.15).

Ao ser executado o *rsync* (linha 8), agora com o acréscimo da opção “*--progress*”, podemos acompanhar a transferência apenas do arquivo mais recente entre as pastas (linha 10). Nas linhas 13 a 17 podem ser visualizados os arquivos na pasta de destino da sincronização.

Quadro 8.16 – Sincronizando pastas usando *rsync*

```
1. # ls -l livro
2. -rw-r--r-- 1 root root 1095 Out 9 2015 bkp.bat
3. -rw-r--r-- 1 root root 216 Nov 15 08:13 capitulo8.txt
4. -rw-r--r-- 1 root root 332 Nov 14 18:53 HarryPotter.txt
5. -rw-r--r-- 1 root root 492 Nov 14 18:54 tcp.png
6. -rw-r--r-- 1 nobody nogroup 102 Nov 14 20:10 teste.txt
7.
8. # rsync -a --progress livro/ livro-bkp/
9. sending incremental file list
10. capitulo8.txt
11.          216 100%    0.00kB/s   0:00:00 (xfr#1, to-chk=2/6)
12.
13. # ls -l livro-bkp/
14. -rw-r--r-- 1 root root 1095 Out 9 2015 bkp.bat
15. -rw-r--r-- 1 root root 216 Nov 15 08:13 capitulo8.txt
16. -rw-r--r-- 1 root root 332 Nov 14 18:53 HarryPotter.txt
17. -rw-r--r-- 1 root root 492 Nov 14 18:54 tcp.png
-rw-r--r-- 1 nobody nogroup 102 Nov 14 20:10 teste.txt
```

Fonte: Elaborado pelo autor.

O comando *rsync* ainda tem mais opções, podendo por exemplo, pular a cópia de determinado tipo de arquivo, comprimir as informações nas transferências (útil em grandes transferências via internet) ou limitar os arquivos a serem transferidos pelo tamanho.

### 8.6.3 SCP (Secure Copy)

O utilitário *scp* faz parte do pacote *ssh* já comentado em capítulos anteriores. Ele se utiliza do túnel criptografado do *ssh* para executar a cópia de arquivos entre dois dispositivos remotos.

Quadro 8.17 – Copiando arquivos remotos com *scp*

```
1. # scp -r -p root@192.168.0.12:/root/sistema/banco-
   -dados/ /var/bkp

2. root@192.168.0.12's password:

3. base.dbf          100% 14KB 14.0KB/s 00:00
4. base.idx          100% 405    0.4KB/s 00:00
```

Fonte: Elaborado pelo autor.

No quadro 8.17, foi feita uma cópia recursiva (“*-r*”) indicando que todos os arquivos e pastas da origem serão copiados e serão preservadas as propriedades dos arquivos (por exemplo, data, hora, permissões). Estes arquivos serão copiados para a pasta local “*/var/bkp*”. É informado para o comando o nome de usuário (“*root*” separado por um @) da máquina de destino que será usado para estabelecer a conexão e identificar as permissões de acesso.

Durante a cópia, todas as informações passam criptografadas e não é possível verificar o conteúdo utilizando um “*tcpdump*” ou “*wireshark*”. Para automatizar algumas transferências onde o usuário agendará a cópia e não pode ficar fornecendo usuário e senha, é possível o uso de uma chave pública (mecanismo de segurança) para o acesso ao serviço.

## Síntese

Neste oitavo capítulo, pode ser verificado o uso da transferência e compartilhamento de arquivos entre dispositivos da rede. Pode ser acompanhado o contexto das redes locais e a evolução das tecnologias de compartilhamento de arquivos. Os principais protocolos de compartilhamento de arquivos SMB e NFS foram analisados e exemplificados, assim como exemplificada a inte-

roperação possível entre as redes dos dois principais sistemas corporativos (Windows e Linux). O uso de armazenagem de dados na rede local e na nuvem foi apresentado, com a utilização prática de um serviço de nuvem para armazenamento de dados corporativos. O uso dos utilitários, como o *aws*, *rsync* e *scp*, podem automatizar as transferências de arquivos entre servidores em uma rede.

## Atividades

1. Marque com V (verdadeiro) e F (falso) as afirmações a seguir sobre o compartilhamento de arquivos na rede usando o SMB.
  - ( ) é possível acessar um compartilhamento de pasta de um sistema Windows dentro de um sistema Linux.
  - ( ) o uso de compartilhamento de arquivos via Samba no Linux permite apenas a leitura de arquivos do Windows pois o sistema de arquivos é diferente entre os dois sistemas.
  - ( ) o Windows consegue verificar os compartilhamentos apenas na interface gráfica, usando o gerenciador de arquivos “explorer”.
  - ( ) o protocolo SMB pode ser utilizado para imprimir remotamente em impressoras.
2. No protocolo NFS explique como fica a montagem do compartilhamento na estrutura de arquivos do cliente.
3. Qual a diferença entre SAN (*Storage Area Network*) e NAS (*Network Attached Storage*)?
4. Explique o conceito de sincronia de arquivos utilizado nos comandos “aws” e “rsync”?

# 9

## A Empresa e Seus Desafios

NESTE CAPÍTULO, ESTUDAREMOS as redes corporativas e os desafios técnicos que exigem constante monitoramento e busca de novas soluções tecnológicas por parte do profissional de redes. A quantidade de aspectos que precisam ser cuidados para que a rede de uma empresa tenha o menor índice de **interrupção** estimula o profissional a pesquisar ferramentas, métodos, tecnologias ou novos processos que possam auxiliar nessa busca por maior estabilidade. A evolução e a adaptação das empresas a um mercado cada vez mais exigente impõem uma maior flexibilidade no uso dos recursos.

As TECNOLOGIAS QUE envolvem a virtualização de ambientes tanto interna quanto externamente vêm sendo consideradas uma das respostas para essa maior flexibilidade na gestão da rede corporativa.

## 9.1 Rede corporativa

Uma empresa, conforme seu porte, necessita de algum tipo de tecnologia para sustentar seus processos de serviço ou de industrialização. Mesmo empresas muito pequenas podem precisar de apoio da internet e de máquinas de pagamento de cartão que fazem acesso externo ou pequenos roteadores sem fio para o acesso de um computador usado na administração da empresa.

Mas, à medida que a empresa se expande (ou as que já iniciam com maior porte), adota-se uma lógica recorrente associada ao investimento em tecnologia. Ao aumentar a quantidade de colaboradores, aumenta também a necessidade de máquinas. Isso leva à necessidade de as máquinas se interligarem para trocar arquivos. Ao configurar e expandir a rede local, considera-se também a tendência de centralizar os arquivos para evitar redundância e falta de controle e facilitar as cópias de segurança. Com essa situação, o uso de um servidor dedicado e que auxilie no compartilhamento torna-se viável, e quando o volume de acesso e serviços no servidor aumenta, multiplicam-se a instalação e a configuração de mais servidores na rede.

Um equipamento de informática, independentemente do tipo, pode ter algum tipo de manutenção que suspenda seu funcionamento, variando a frequência dessa manutenção conforme a qualidade de seu hardware e das condições do ambiente em que fica em funcionamento. Ambientes com energia elétrica controlada (*no-breaks*), refrigeração e pouca intervenção humana diminuem as possibilidades de interrupção dos serviços e de seus servidores. Essas situações devem ser monitoradas frequentemente, com métricas para avaliação da qualidade dos dispositivos em uso que devem orientar o profissional de rede sobre quais ações devem ser planejadas para melhorar o desempenho ou diminuir o percentual de indisponibilidade dos serviços. Em alguns casos, essas medidas podem ser negociadas com a empresa para adequar a expectativa que os processos de negócio têm da tecnologia.

Todo esse cenário impõe constantes avaliações de quanto investir na expansão e na manutenção dessa estrutura. Mesmo as empresas de volume de faturamento semelhante podem ter redes de computadores bem diferentes: com segmentação da rede em andares ou filiais, diferentes sistemas administrativos sendo executados em mais de um servidor, maior ou menor

criticidade com relação ao acesso à internet ou volume variável de arquivos manipulados através da rede.

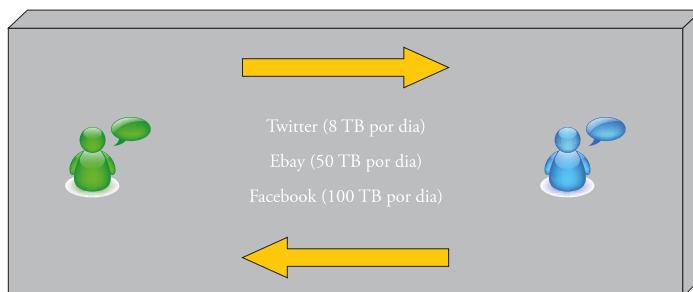
O custo de manutenção do parque tecnológico é bem abrangente, pode variar de limpeza periódica dos terminais das baterias dos *no-breaks*, passando por situações eventuais, como troca de fontes de servidores e troca de placas de alta velocidade em *switches* centrais, indo para pagamento de licenças de sistemas e conexões à internet e filiais.

Por esse motivo, uma empresa está periodicamente acompanhando e avaliando novas tecnologias que proporcionem melhor uso da infraestrutura ou novas formas de aumentar seu faturamento.

### 9.1.1 Volume de dados

A internet tem possibilitado a circulação de um volume muito grande de dados e isso tem exigido da tecnologia uma rápida evolução no desempenho e na capacidade das redes. Na Figura 9.1, o volume de tráfego de alguns serviços durante o dia revela a dimensão em que alguns roteadores na internet devem trabalhar.

Figura 9.1 – Volume de dados manipulados por dia



Fonte: Elaborado pelo autor com base em Lytle (2013).

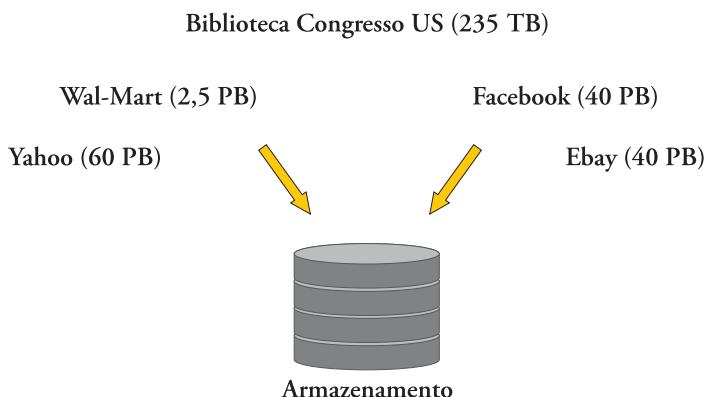
Enquanto a transmissão de dados via internet revela que tipo de capacidade de rede e processamento precisa ser considerada em alguns serviços de maior abrangência, em algum momento essas informações precisam ser armazenadas para posterior análise e visualização. Esse armazenamento (como

## Rede de Computadores

visto no capítulo 8), precisa estar compartilhado na rede para ser acessado por servidores e usuários do serviço.

O armazenamento dos dados nas empresas tem crescido (ver Figura 9.2) à medida que precisam ser registradas diversas ações dos usuários dos serviços, e a quantidade de usuários vem aumentando consistentemente. Mas é preciso considerar que, além da geração humana de dados, é necessário planejar o volume de armazenamento considerando a geração de dados por outros equipamentos.

Figura 9.2 – Volume de dados armazenados



Fonte: Elaborado pelo autor com base em Lytle (2013).

A coleta automática de dados de sensores por computador aumenta ainda mais a necessidade de planejamento de capacidade de armazenamento nas empresas.

---

### Para refletir...

O uso de sensores em aviões atualmente proporciona um conjunto de dados que auxilia a manutenção e a análise de diversas condições que a aeronave precisa enfrentar. Um dos maiores fabricantes de aviões do mundo instala, em cada avião fabricado, entre 8 mil e 10 mil sensores. Um Boeing

737 produz 40 terabytes de informação por hora durante o voo. Se considerarmos a quantidade de aviões comerciais em circulação em um dia nos Estados Unidos (aproximadamente 28,5 mil), podemos ter uma perspectiva do que está acontecendo com algumas empresas em relação ao volume de dados em circulação. Esses dados, sendo segmentados, compartilhados para diferentes setores dentro da empresa ou analisados em ferramentas de pesquisa de dados demandam servidores e consequentemente redes com grande capacidade.

Ao comparar os demais exemplos de volume transmitidos e armazenados em algumas empresas exemplificados neste capítulo com os novos tipos de cabeamento sendo desenvolvidos (vistos no Capítulo 1) e com a disseminação de maiores velocidades da tecnologia Ethernet usada em armazenagem de dados em rede (vistos no Capítulo 8), podemos avaliar que ainda é preciso evoluir bastante no desempenho e na capacidade das redes para enfrentar esse crescimento. E profissionais capacitados em rede e tecnologias de armazenamento e processamento distribuído serão bem valorizados no mercado de trabalho..

---

### 9.1.2 Novos dispositivos

O uso de dispositivos pessoais tem aumentado e pode revelar uma série de características no perfil de uso dos serviços e auxiliar a empresa a melhorar a personalização da comunicação ou da prestação de serviço. Uma série de informações está sendo coletada em *smartphones*, como posicionamento, velocidade desenvolvida, trajeto percorrido, assuntos pesquisados e contatos. Também estão sendo centralizadas nos *smartphones* informações vindas de sensores que podem ser inseridos em roupas e acessórios, os quais se comunicam diretamente pela rede e são chamados de IoT (*Internet of things* – Internet das coisas). Tênis, jaquetas, relógios (*smartwatches*), dispositivos que ficam

dentro do organismo liberando remédio e óculos de realidade aumentada são alguns dos dispositivos externos que, em muitos casos, centralizam as informações no *smartphone* tornando-o um “servidor pessoal”.

No momento que o usuário começa a trazer esses dispositivos para dentro da empresa, ela precisa se reorganizar com relação à gestão dos ativos de tecnologia e seu departamento de TI deve começar a incorporar novos sistemas no suporte do dia a dia. Por exemplo, é comum ter *smartTV* em salas de reunião para uso em apresentações, diretores têm *tablets* para acessar dados de sistemas administrativos e de sistemas de BI (*Business Intelligence*), supervisores têm *smartphones* corporativos para mandar fotos de um problema na fábrica, gerentes enviam mensagens instantâneas por celular solicitando informações no meio de uma reunião.

Do ponto de vista do usuário, é apenas mais uma forma de acesso a informações para uso em suas atividades (ou entretenimento). Mas, para a empresa, as preocupações e os cuidados se multiplicam. As conexões à internet originadas nas atualizações de vários dispositivos adicionais na rede, por exemplo, competem com o acesso normal dos demais servidores e computadores da empresa. E, com tantos serviços críticos e dispositivos conectados, o ambiente no qual os servidores e as unidades de armazenamento estão instalados necessitam cada vez de maior cuidado e investimento, de maneira que, se houver algum problema grave no equipamento, seja possível rapidamente substituir ou acionar medidas de contingência.

Uma alternativa que vem sendo utilizada por várias empresas para evitar o investimento e a manutenção de ambientes complexos para os servidores é o uso de infraestrutura externa (servidores e *storages*), com a possibilidade de pagar apenas o que for utilizado desses equipamentos.

## 9.2 Computação em nuvem

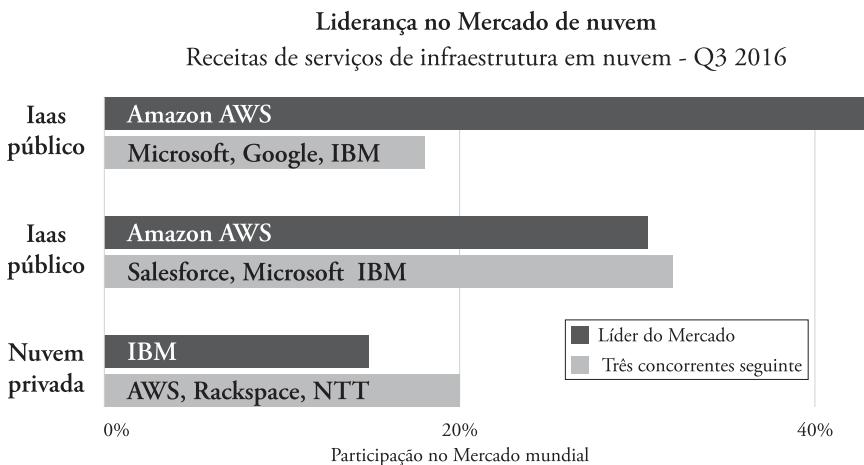
No capítulo 8, foi verificado o uso de um serviço de armazenamento em nuvem utilizando o provedor Amazon como exemplo. Mas o serviço de computação em nuvem é bem mais abrangente.

Conforme o provedor, pode-se configurar um servidor virtual com opções de quantidade de processador, memória, volumes de armazenamento

e também tecnologias de alta disponibilidade, sistemas de *backup* e filtragens de segurança. Além de servidores e armazenagem de arquivos, outros serviços específicos, como ambiente de análise de dados, bases de dados no SQL, gerenciamento de fluxo de dados em tempo real, centralização de dados de sensores e *desktops* virtuais podem ser contratados sem as preocupações relacionadas à manutenção dos equipamentos, a energia elétrica estabilizada, a refrigeração e as conexões entre dispositivos. O mercado para esse tipo de serviço tem crescido e muitas grandes empresas de informática estão se capacitando para oferecer opções de diferentes níveis de tecnologia.

Para exemplificar algumas funcionalidades, usaremos o provedor Amazon com seu serviço de WEB (AWS – *Amazon Web Service*) por ser a principal empresa nesse mercado. Sua capacidade de computação e volume de recursos supera as demais empresas que prestam o mesmo tipo de serviço (ver figura 9.3).

**Figura 9.3 – Mercado de infraestrutura em nuvem**



Na figura 9.3, o primeiro indicador mostra o tamanho do mercado de serviço de infraestrutura pública na nuvem, com a Amazon com o dobro do tamanho dos três fornecedores seguintes somados (Microsoft, Google e IBM). Também há serviços de infraestrutura que podem ser contratados de modo privado, ou seja, a empresa contratante controla e gerencia os dispositivos.

tivos e não compartilha o ambiente com outras empresas. Nesse caso, a IBM tem sido a principal fornecedora de infraestrutura privada.

Além de fornecer uma estrutura virtual para ser configurada e utilizada remotamente, outros tipos de serviço em nuvem podem ser oferecidos pelos provedores. No quadro 9.1 podem ser verificados como são classificados os tipos de serviço em nuvem.

Quadro 9.1 – Tipos de serviço em nuvem

<b>IaaS (<i>Infrastructure as a Service</i> – Infraestrutura como um serviço)</b>	O usuário tem acesso a um sistema operacional no qual pode instalar e gerenciar suas aplicações. Toda a infraestrutura física fica sob os cuidados do provedor.
<b>PaaS (<i>Platform as a Service</i> – Plataforma como um serviço)</b>	O provedor fornece um ambiente (bibliotecas de desenvolvimento, linguagem de programação, utilitários) e o usuário pode desenvolver ou instalar aplicações compatíveis com o ambiente. O servidor, o sistema operacional e os dispositivos de rede e armazenamento são controlados pelo provedor.
<b>SaaS (<i>Software as a Service</i> – Software como um serviço)</b>	Acesso direto a aplicação remotamente. O usuário não controla a infraestrutura (servidor, sistema operacional, rede ou hardware de armazenamento) na qual o sistema está instalado.

Fonte: Elaborado pelo o autor com base em Mell; Grance (2008).

No caso do SaaS, o sistema aplicativo é acessado diretamente no provedor. Uma empresa pode fornecer o uso de seu sistema aos clientes ou usuários de forma remota, sem se preocupar onde ele está instalado. Assim tem acontecido, por exemplo, com pacotes de escritórios acessados diretamente na web (Office 365, Google Docs) e sistemas administrativos (ContaAzul, MarkUp).

O IaaS permite maior flexibilidade e controle, uma vez que, a partir da escolha do sistema operacional, todas as aplicações, os utilitários e a gestão dos recursos ficam sob sua responsabilidade. O serviço da Amazon que possibilita contratar um servidor com essas características é o EC2.

### 9.2.1 Amazon EC2

No Capítulo 8 foi apresentado um serviço de armazenamento (*storage*) na nuvem no qual os arquivos eram armazenados e recuperados pela internet. Outro serviço desse provedor é o EC2, que permite escolher um servidor virtual com um tipo de capacidade e sistema operacional e, dessa forma, habilitar o ambiente para uso em nuvem. Esses servidores no serviço da Amazon são referenciados como instâncias (*instances*).

Para iniciar o uso do servidor, é necessário escolher o sistema operacional disponível no provedor (ver Figura 9.4). Vários tipos de sistemas operacionais estão disponíveis e alguns podem ser utilizados gratuitamente (*free tier eligible*) por 12 meses, conforme a capacidade do servidor selecionado.

Figura 9.4 – Escolha do sistema operacional



Fonte: AWS (2016).

A partir da seleção do sistema operacional, é feita a seleção da quantidade de processadores e memória. Também são escolhidos os tamanhos dos discos (em bytes) e o tipo de tecnologia de armazenamento: magnético ou SSD (*Solid State Drive*).

A Amazon tem uma denominação própria para algumas combinações pré-montadas de processador e memória que podem ser vistas na coluna *Type*

da Figura 9.5. Cada nome indica um tipo de uso: baixo uso (T2), uso geral (M3/M4), foco em processamento (C3/C4), uso intensivo de memória (X1/R3), processamento matemático (P2/G2) e foco em espaço em disco (I2/D2). Na Figura 9.5, algumas combinações (T2 e M4) podem ser verificadas com as opções de processadores e memória.

Figura 9.5 – Escolha da capacidade

	Family	Type	vCPUs	Memory (GiB)
<input type="checkbox"/>	General purpose	t2.nano	1	0.5
<input checked="" type="checkbox"/>	General purpose	t2.micro <small>Free tier eligible</small>	1	1
<input type="checkbox"/>	General purpose	t2.small	1	2
<input type="checkbox"/>	General purpose	t2.medium	2	4
<input type="checkbox"/>	General purpose	t2.large	2	8
<input type="checkbox"/>	General purpose	m4.large	2	8
<input type="checkbox"/>	General purpose	m4.xlarge	4	16
<input type="checkbox"/>	General purpose	m4.2xlarge	8	32
<input type="checkbox"/>	General purpose	m4.4xlarge	16	64
<input type="checkbox"/>	General purpose	m4.10xlarge	40	160

Fonte: AWS (2016).

Cada servidor deve estar associado a um disco para armazenamento de informações e do sistema operacional, mas também podem ser anexados mais discos de capacidades variadas para uso no servidor. Após a criação do volume, o sistema operacional identifica o novo espaço definido como um disco adicional e pode ser formatado e associado à estrutura de arquivos. No Capítulo 8, foram verificadas algumas diferenças na maneira de anexar um recurso de armazenamento a uma estrutura de arquivos nos sistemas Win-

dows e Linux. Na Figura 9.6, temos alguns exemplos de volumes de discos com diferentes capacidades (de 30 gigabytes a 1,5 terabytes).

Figura 9.6 – Volumes de discos virtuais

Name	Volume ID	Size	Volume Type	IOPS	Snapshot	Created
Portal	vol-05ew20e5	30 GiB	gp2	100 / 3000	snap-0d9fc25b	May 7, 2015 at 5:48...
Disco C V12	vol-2acd1fa2	1500 GiB	gp2	4500	snap-05c6987c4700208	September 27, 2016...
	vol-cf99e97b	1000 GiB	gp2	3000	snap-0f8aa80a	November 3, 2016 at...
	vol-cf88e972	30 GiB	gp2	100 / 3000	snap-0b51eef73200a7a1a	November 3, 2016 at...
Arquivos Co...	vol-8cf11674	100 GiB	gp2	300 / 3000		November 8, 2016 at...

Fonte: AWS (2016).

Nos volumes da figura 9.6, a coluna *Name* possibilita criar uma identificação para o disco e facilitar o gerenciamento. A coluna *Snapshot* indica um recurso utilizado para criar uma cópia da situação do disco naquele momento, usado para criar cópias de segurança periódicas ou antes de alguma intervenção.

Diversas opções de suporte aos servidores também estão disponíveis através da linha de comando usando o utilitário aws, da mesma forma que foi utilizado no Capítulo 8, quando foram verificadas as informações no *storage* remoto. No Quadro 9.2, vemos a sintaxe do utilitário aws para listar todas as instâncias que foram criadas em um contrato.

Quadro 9.2 – Listando as informações das instâncias (AWS)

```
aws --output "table" --region "us-west-2" ec2
describe-instances
```

Fonte: Elaborado pelo autor.

Neste exemplo, foi utilizada a opção “--output ‘table’” para indicar a forma de visualização do resultado. Podem ainda ser utilizados os formatos “text” e “json”. Todas as opções do comando “aws” podem ser listadas na tela ao utilizar “aws help”. Foi indicada no exemplo também a região na qual os servidores estão configurados, “--region ‘us-west-2’”, e para qual tipo de ser-

viço será executada essa operação (neste caso, “ec2”). Por último, é solicitada a descrição de todas as instâncias configuradas em “describe-instances”.

No Quadro 9.3, temos uma parte das informações retornadas pelo comando “aws”. No trecho mostrado estão informações sobre uma das instâncias configuradas com o nome chave de “LIVRO”.

Quadro 9.3 – Informação das instâncias em formato “table” (AWS)

DescribeInstances	
Reservations	
OwnerId	86xxxxxxxxx830
ReservationId	r-90xxx59a
Instances	
AmiLaunchIndex	0
Architecture	x86_64
ClientToken	ignd14xxxxxxxxxx
EbsOptimized	False
Hyperervisor	xen
ImageId	ami-5189a661
InstanceId	i-ca2c8f3d
InstanceType	t2.micro
KeyName	LIVRO
LaunchTime	2015-05-07T20:48:02.000z
PrivateDnsName	ip-172-16-18-38.us-west-2.compute.internal
PrivateIpAddress	172.16.18.38
PublicDnsName	ec2-52-XX-XXX-245.us-west-2.compute.amazonaws.com
PublicIpAddress	52.XX.XX.45
RootDeviceName	/dev/sdal
RootDeviceType	ebs
SourceDestCheck	True
StateTransitionReason	None
SubnetId	subnet-b74bd3d2
VirtualizationType	hvm
VpcId	vpc-b966xxxx
BlockDeviceMappings	
DeviceName	/dev/sdal
Ebs	
AttachTime	2015-05-07T20:48:05.000z
DeleteOnTermination	True
Status	attached
VolumeId	vol-f5xxxxe5

Fonte: Elaborado pelo autor.

Nessa descrição também podemos verificar o tipo de sistema utilizado pela identificação da “imagem” (termo usado para uma cópia da instalação de um sistema operacional). Cada sistema operacional listado como opção tem

um identificador. Nesse caso, “ami-5189a661” indica um servidor “Linux Ubuntu 14.04”.

#### Quadro 9.4 – Acessando servidor virtual (AWS)

```
ssh -i LIVRO.pem ubuntu@52.12.158.245
Welcome to Ubuntu 14.04.2 LTS (GNU/Linux 3.13.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
 
 System information as of Mon Oct 24 16:33:51 BRST 2016

 System load:  0.0          Processes:           113
 Usage of /:   4.8% of 29.39GB  Users logged in:    0
 Memory usage: 21%          IP address for eth0: 172.16.18.38
 Swap usage:   0%
 
 Graph this data and manage this system at:
 http://landscape.canonical.com/
 
 Get cloud support with Ubuntu Advantage Cloud Guest:
 http://www.ubuntu.com/business/services/cloud

203 packages can be updated,
126 updates are security updates.

Last login: Mon Oct 24 16:33:53 2016 from 17.99.10.11
ubuntu@ip-172-16-18-38:~$
```

Fonte: Elaborado pelo autor.

No quadro 9.4 há um exemplo de acesso ao servidor virtual em execução na Amazon EC2. Foi utilizado o comando “ssh” já mostrado em outros capítulos, mas dessa vez com a opção “-i LIVRO.pem”, que é a forma padrão de acesso a servidores Linux dentro da Amazon, onde o serviço EC2 cria uma chave de acesso como alternativa para o uso de senha do usuário. Para esse tipo de sistema é usado como usuário padrão o nome “ubuntu”. Em seguida, as mensagens iniciais do acesso ao servidor, com um resumo dos processos em execução e os recursos utilizados.

Para os servidores do tipo *Windows Server*, é criado um acesso via *Terminal Service*, que permite o uso da interface gráfica para administração remota do servidor.

## 9.2.2 Amazon WorkSpace

Com a maior diversidade de dispositivos sendo utilizados nas empresas, aumentam os tipos de integração e as formas de acessos às aplicações corporativas. A constante mobilidade dos colaboradores, o uso de colaboradores temporários ou terceirizados e a necessidade de atualização periódica dos aplicativos em uso exigem um esforço do setor de TI das empresas para dimensionar corretamente a frequência da atualização das aplicações, a incompatibilidade entre diferentes aplicativos, o volume de cópias de segurança e a capacidade das redes e dos servidores.

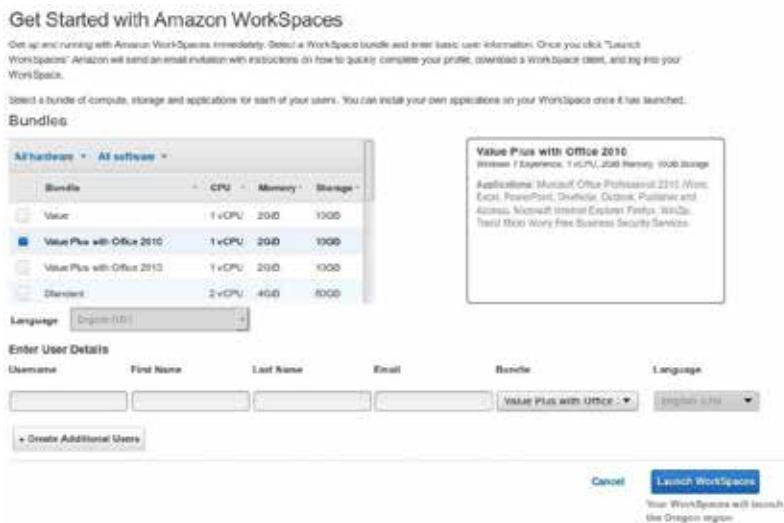
Da mesma maneira que a virtualização e o uso remoto de servidores têm proporcionado uma alternativa para a administração e o gerenciamento desses equipamentos, o uso de *desktops* virtuais e remotos pode ser uma possibilidade para a empresa. O serviço Workspace da Amazon permite criar um sistema *desktop* que será utilizado por dispositivos com diferentes ambientes, como Mac OS, iPad, Windows, Android Tablet, ChromeOS, Fire Tablet e acesso web (browser).

### Saiba mais

Com o aumento no desenvolvimento e no uso de vários tipos de sistemas em formato web, um novo tipo de conceito em sistemas operacionais começou a ser considerado: o *browserOS* (sistema operacional baseado em *browser*). Esse tipo de sistema operacional se baseia no ambiente de execução de aplicações web do *browser* e uma camada enxuta de código para reconhecer o *hardware* e iniciar o navegador. Dois sistemas estão disponíveis nesse formato: ChromeOS e FirefoxOS. Enquanto o FirefoxOS tem sido utilizado em *smartTVs*, diversos fabricantes de *notebooks* têm criado uma linha para o ChromeOS - os ChromeBooks.

Na figura 9.7, temos um exemplo de configuração de *desktop* com aplicativos adicionais (como Office 2010) e com o dimensionamento de memória e quantidade de processadores a serem usados.

Figura 9.7 – Forma de acompanhamento de uso das instâncias de servidores (Amazon)



Fonte: Elaborado pelo autor.

Após a inicialização do *desktop*, podem ser instalados outros aplicativos específicos da empresa e o ambiente estará pronto para ser acessado pelos sistemas indicados anteriormente.

### 9.2.3 Acompanhamento da demanda

Uma das características relacionadas ao uso de infraestrutura em nuvem é a possibilidade de pagar apenas o tempo que o serviço foi utilizado. Nos serviços da Amazon, cada tipo de funcionalidade é cobrado por tempo de funcionamento. Também são avaliados os volumes de transferência para cada tipo de armazenagem e os serviços adicionais de monitoramento e alerta. Na Figura 9.8 há um detalhamento dos serviços utilizados de servidores EC2 com o volume de dados de disco utilizado pelos servidores. Na parte superior está o servidor Linux que esteve em execução durante o mês em um servidor tipo “t2-micro”. Depois, dois servidores Windows em um servidor “m4-2xlarge” e “r3-4xlarge”. A seguir, está o detalhamento do uso dos volumes de armazenamento anexados aos servidores.

Figura 9.8 – Painel de acompanhamento de uso das instâncias de servidores (Amazon)

US West (Oregon) Region			
Amazon Elastic Compute Cloud running Linux t2.micro	\$0.015 per On-Demand Linux t2.micro Instance Hour	729 Hrs	\$9.49
Total:			\$9.49
Amazon Elastic Compute Cloud running Windows	\$0.060 per On-Demand Windows m1.4xlarge Instance Hour	729 Hrs	\$310.61
	\$1.344 per On-Demand Windows x3.4large Instance Hour	729 Hrs	\$1,417.18
Total:			\$1,417.18
EBS			
\$0.000 per 1000 MiBps per m1.2xlarge instance-hour (per journal hour)		729 Hrs	\$0.00
\$0.05 per 1 million I/O requests - US West (Oregon)		41,321,819 I/Os	\$2.06
\$0.05 per 1GB-month of storage (per month) Storage - US West (Oregon)		974,463 GB-Mo	\$48.72
\$0.05 per 1GB-month of snapshots (per month) - US West (Oregon)		102,150 GB-Mo	\$5.11
\$0.10 per 1GB-month of General Purpose SSD (gp2) provisioned storage - US West (Oregon)		1,430,917 GB-Mo	\$143.09
Total:			\$205.88
Region Total:			\$1,649.59

Fonte: Elaborado pelo autor.

Caso haja utilização de outros serviços, como o S3 visto no capítulo anterior, são apresentados nesse painel o detalhamento de uso e transferências de dados.

## 9.3 Virtualização interna

O uso de servidores virtuais na nuvem traz algumas facilidades no gerenciamento e na manutenção dos servidores, pois podem ser copiados, duplicados e utilizados como base para outras instalações, mas para ter essas facilidades na empresa não é necessário contratar um provedor externo, já que sistemas de virtualização de servidores podem ser utilizados gratuitamente. Dois dos principais sistemas são Vmware (proprietário, mas com versão gratuita) e XEN (*opensource*). Será usado o sistema XEN como referência, que é utilizado em serviços de nuvem como o EC2 da Amazon e de outros grandes fornecedores.

### Saiba mais

Em termos de virtualização, existem dois tipos principais: virtualização completa e virtualização em sistema operacional. No primeiro caso,

o código de virtualização (*hypervisor*) é executado diretamente no hardware e, no segundo caso, um sistema operacional é instalado no hardware e depois é executado um sistema de virtualização. Uma alternativa chamada paravirtualização se aproveita de alterações no código do sistema cliente para acessar direto o hardware disponibilizado pelo *hypervisor*. O XEN está associado ao primeiro tipo de virtualização e o VirtualBox, sugerido no primeiro capítulo, ao segundo tipo.



### 9.3.1 Projeto XEN

O sistema XEN foi criado no início da década de 1990 como um projeto de pesquisa da Universidade de Cambridge. Em 2002, foi tornado *opensource* e tem sido incorporado como opção em vários serviços de virtualização de grande porte e incluído como opção de virtualização de servidores em sistemas Linux. Os pesquisadores do projeto original fundaram uma empresa para comercializar o produto, que foi comprada pela Citrix, a qual tem mantido suporte ao projeto *opensource*. Em 2013, o projeto *opensource* tornou-se parte da Linux Fundation e recebeu apoio de várias empresas, incluindo Alibaba, Amazon, AMD, Citrix, Huawei, Intel e Oracle.

Um sistema completo, desenvolvido pela Citrix, pode ser acessado no site <[www.xenserver.org](http://www.xenserver.org)> e, após ser instalado, utiliza um aplicativo cliente para o gerenciamento e a manutenção do servidor, assim como na criação dos servidores virtuais. Como acontece no mundo *opensource*, é possível instalar o sistema sem pagar licenciamento, mas podem ser contratados suporte e customizações no sistema por meio de empresas de tecnologia. A possibilidade de verificar o código fonte facilita a adaptação do sistema ao modelo de negócio (como fez a Amazon em seu EC2).

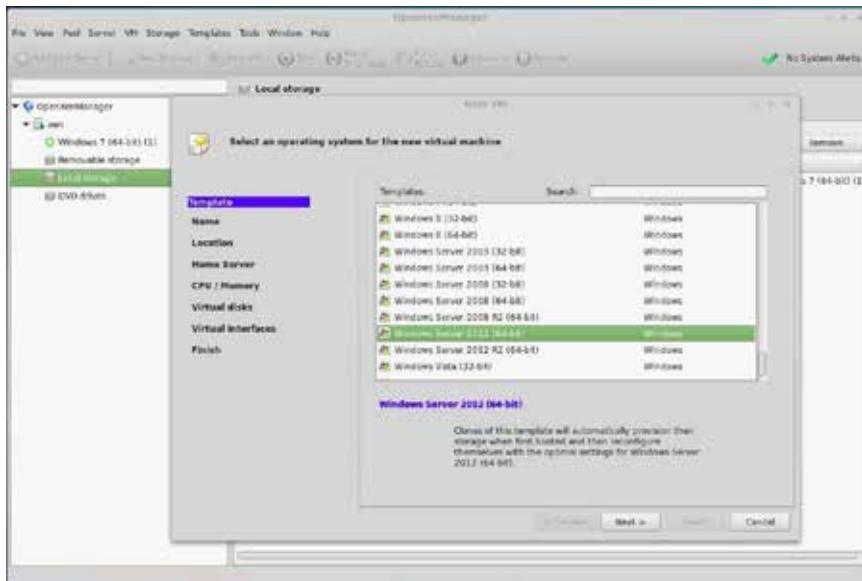
#### 9.3.1.1 Criação de um servidor virtual

O processo de criação do novo servidor é semelhante ao utilizado na criação do servidor em nuvem (EC2). Seleciona-se a quantidade de memória e processadores e identifica-se o sistema operacional a ser instalado, para que alguns parâmetros sejam inicializados conforme o sistema escolhido. Mas a

instalação do sistema operacional envolve o uso das mídias de instalação do fabricante e o licenciamento nos casos de sistemas operacionais proprietários.

Na figura 9.9, vemos um exemplo de criação de uma VM (*Virtual Machine* – Máquina Virtual) utilizando o aplicativo OpenXenManager. Deve ter sido escolhida previamente a mídia na qual será instalado o servidor.

Figura 9.9 – Tela de criação de um servidor virtual (XEN)

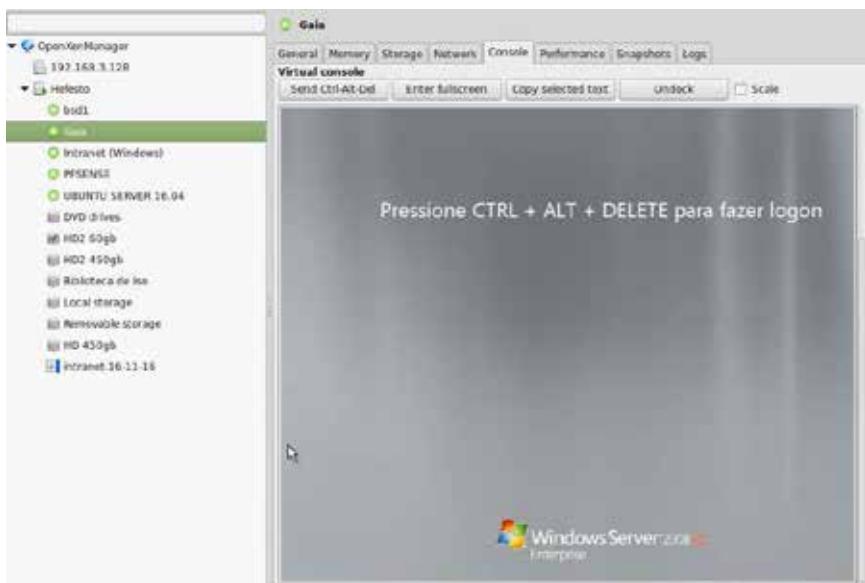


Fonte: Elaborado pelo autor.

Após ter sido instalado o sistema operacional, pode-se acessar o console do equipamento virtual da mesma maneira que um servidor físico. Nesse console podem ser feitas manutenções emergenciais quando o acesso via rede não for possível.

Na Figura 9.10, no lado direito do aplicativo está a tela inicial do servidor Windows Server 2008 instalado em uma VM no XEN. No lado esquerdo, as demais máquinas virtuais instaladas que estão em execução (identificadas pelo símbolo em verde ao lado do nome da máquina).

Figura 9.10 – Acesso ao console do sistema instalado (XEN)



Fonte: Elaborado pelo autor.

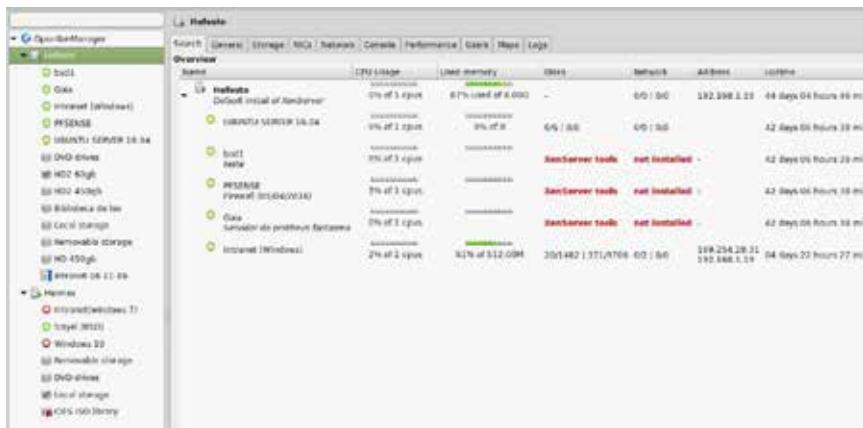
Ao instalar vários equipamentos com o sistema XEN, o aplicativo de gerenciamento permite que sejam conectados e visualizados os ambientes de cada instalação com as máquinas virtuais clientes de cada um. Com isso, é possível também verificar a situação dos recursos utilizados por cada ambiente. Para saber informações detalhadas de cada cliente virtual, deve ser instalado um utilitário (*xen server tools*) que coleta as informações de uso dos recursos e informa no painel central.

Na figura 9.11, no lado esquerdo estão listados dois ambientes XEN, com suas VM e cada uma com diferentes sistemas operacionais. No lado direito está o uso geral dos recursos do equipamento físico mostrado no topo e para cada VM mostrado o recurso em uso naquele momento. Esses recursos da VM serão mostrados apenas se a máquina cliente tiver o utilitário comentado instalado. No exemplo, a última VM mostrada no painel (lado direito) é uma máquina Windows 7 com o utilitário XEN instalado, mostrando as

## Rede de Computadores

informações da memória em uso, o espaço utilizado, o espaço total de cada disco e quantos IPs estão associados à VM.

Figura 9.11 – Acompanhamento de uso dos recursos (XEN)



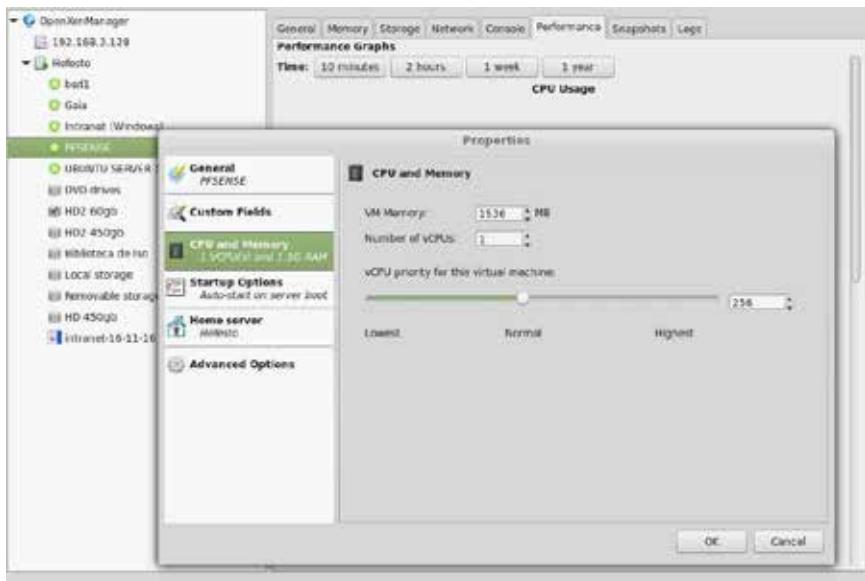
Fonte: Elaborado pelo autor.

Na figura, no lado direito do painel podem ser verificadas na parte superior as abas de informações específicas. Ao selecionar o servidor no lado esquerdo, no lado direito são mostradas as informações daquele servidor ou do ambiente XEN. Pelas abas podem ser verificadas a situação dos discos (*Storage*) e das placas de rede (*NICs*) e as estatísticas de uso (*Performance*). Também podem ser verificadas as informações dos operadores do sistema (*Users*) e os registros da situação do sistema (*Logs*).

Mesmo após ter sido criada a VM e instalado o sistema operacional, podem ser alteradas as características do ambiente, da mesma forma que é possível na máquina física. Nesse caso, o sistema deve ser desligado e depois reiniciado com a nova configuração.

Na Figura 9.12, pode ser vista a janela de propriedades (acionada com o botão direito do mouse com a VM selecionada) e as opções de parâmetros que podem ser revisados.

Figura 9.12 – Verificação de propriedades (XEN)



Fonte: Elaborado pelo autor.

Ao alterar os parâmetros de recursos de memória e os processadores do sistema, é necessário verificar se o servidor físico ainda tem condições de suportar o aumento. Uma das facilidades dos sistemas de virtualização (presente nos servidores em nuvem vistos anteriormente) é a possibilidade de criar um *snapshot* do servidor. Essa opção está disponível na aba de informações específicas ao selecionar uma VM e pode ser utilizada antes de uma grande alteração no sistema e, caso apresente algum problema, volta-se ao estado do *snapshot* criando anteriormente.

Usando o ambiente virtualizado, também é possível duplicar uma VM ou mover uma VM entre dois ambientes XEN. Essa última situação pode ser usada para balancear o uso dos recursos: uma VM pode estar com um processamento muito alto e constante, impactando no desempenho geral do

ambiente. A transferência dessa VM para outro equipamento com menor utilização e com mesma (ou maior) capacidade volta a equalizar a situação.

As conexões de redes entre os servidores são feitas por meio de parâmetros de configuração, uma vez que não há estrutura física a ser conectada. Podem ser criadas sub-redes e utilizados VLANs para fazer a segmentação da rede dos servidores.

## Saiba mais

Alguns fabricantes e provedores estão avaliando arquiteturas de redes baseadas em software: SDN (*Software Defined Networking*).

Essa tecnologia está sendo desenvolvida considerando o estado atual já existente de virtualização de servidores e de conexões de redes virtuais e vem para possibilitar o ajuste via software dos encaminhamentos de rede feitos pelos equipamentos.

## Síntese

Neste capítulo, foram considerados os desafios da administração de uma rede corporativa. A abrangência do escopo da rede e ao mesmo tempo a necessidade de acompanhar os detalhes de cada tecnologia que revelam os desafios do profissional de redes dentro da organização. Foram avaliadas formas de estruturar os serviços de compartilhamento e processamento de dados em um ambiente remoto. Foi exemplificado como pode ser construído um ambiente na nuvem e o acompanhamento do uso do recurso.

Também foram avaliadas maneiras de implementar o uso da virtualização de recursos dentro do ambiente corporativo, mantendo algumas das vantagens proporcionadas pela utilização de recursos remotos como um serviço.

## Atividades

1. Com relação aos tipos de serviço em nuvem, explique qual é a diferença entre IaaS e SaaS.

2. No serviço EC2 da Amazon e na virtualização proporcionada pelo ambiente XEN, qual funcionalidade pode ser usada para voltar a uma situação anterior em caso de problema na manutenção?
3. Cite um dos investimentos feitos no “ambiente de instalação” do servidor que não são necessários caso sejam contratados apenas servidores em nuvem.
4. Considerando os aspectos de computação em nuvem, marque as questões que estão relacionadas a esse conceito.
  - a) A infraestrutura física fica sob responsabilidade do provedor.
  - b) O servidor deve ser enviado para o provedor para instalação em seu ambiente refrigerado.
  - c) Apenas o processamento é feito externamente, os dados devem ser armazenados no disco do *desktop* do usuário.
  - d) Apenas os sistemas do tipo Linux podem ser configurados na nuvem, pois podem ser acessados por SSH.



# 10

## Segurança em Rede

NESTE CAPÍTULO, TRATAREMOS dos aspectos relacionados à proteção da comunicação entre dispositivos. Até aqui, o objetivo dos protocolos e dos conceitos abordados era estabelecer uma conexão e garantir a entrega dos pacotes entre os dispositivos. Neste momento, faremos uma avaliação dos aspectos ligados à restrição da conexão. Também serão abordadas as possibilidades de interceptação da transmissão e os mecanismos de defesa que podem ser incluídos na rede. Protocolos que implementam proteção durante a transmissão e realizam a permissão para o acesso aos compartilhamentos e às informações remotas serão verificados e cuidados adicionais com o uso da internet serão analisados. O tema segurança em computação é bem amplo. Neste capítulo, verificaremos apenas alguns aspectos relacionados ao estudo de redes e forneceremos subsídios para o aprofundamento do tema, caso haja interesse.

## 10.1 Protegendo a rede

No início, os protocolos TCP/IP desenvolvidos para a ARPANET tinham o objetivo principal de transmitir dados entre servidores. Boa parte dos protocolos desenvolvidos utilizavam comandos ASCII, com a informação sendo transmitida sem a preocupação de que a parte de dados pudesse ser inspecionada no meio do caminho.

Com o crescimento do uso e com a arquitetura TCP/IP sendo base tanto para conexões internacionais quanto para a rede local, as informações transmitidas variavam de simples mensagens, acessos a sites de entretenimento ou transmissão de fontes de programas *opensource* para gestão de contas bancárias, uso de dados de bancos de dados privados e transferência de dinheiro virtual. Nem todas as informações podem correr o risco de serem coletadas no meio da transmissão e ter seu conteúdo verificado por pessoas para as quais elas não estão endereçadas. Algumas empresas prestam serviço diretamente via internet e sua sobrevivência depende de que apenas os usuários que efetivamente pagam pelos serviços possam acessar suas funcionalidades.

O estudo deste capítulo estará segmentando à proteção dos recursos compartilhados pela rede em duas partes: a proteção do acesso e a proteção da transmissão. Como a criptografia está na base de várias soluções de segurança, veremos alguns conceitos principais para melhor entender os protocolos que a utilizam.

### 10.1.1 Criptografia

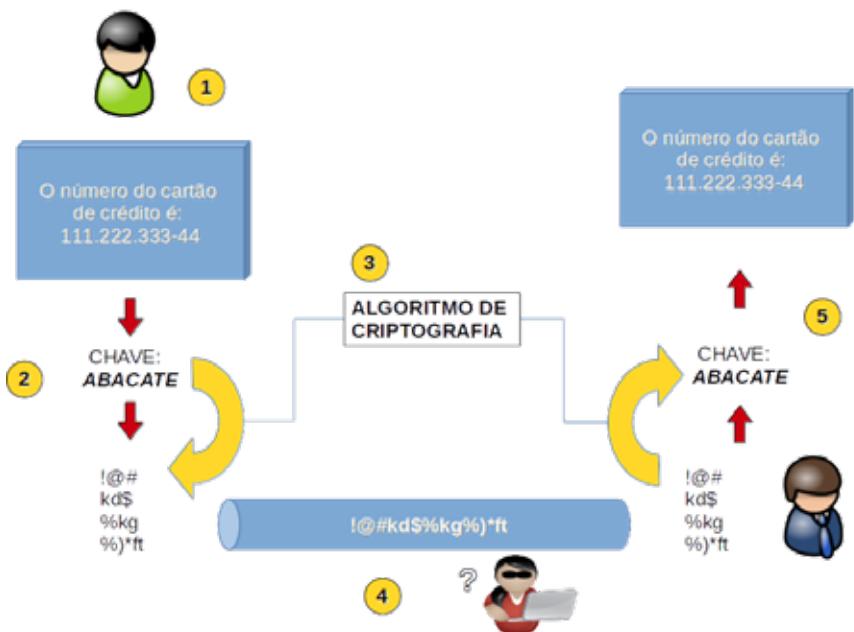
A criptografia se baseia na aplicação de um algoritmo de embaralhamento das informações de maneira que o resultado fique indecifrável. O algoritmo pode ser conhecido e deve ser robusto o suficiente para não ser capaz de reverter o embaralhamento por seu conhecimento e pela posse de qualquer trecho do texto embaralhado.

O resultado do embaralhamento é chamado de texto cifrado. A reversão do texto cifrado no texto original é chamada de descriptografia e o estudo para identificar e reverter o texto cifrado é a criptoanálise.

Existem basicamente dois tipos de criptografia: a criptografia de chave simétrica e a criptografia de chave assimétrica. Na criptografia de chave simé-

trica, a mesma chave é utilizada para criptografar e descriptografar o texto cifrado. Um ponto importante a ser considerado é a maneira como a chave será combinada entre os dois interlocutores, para que um saiba como descriptografar a mensagem do outro. Caso os dois interlocutores estejam distantes, a chave também deve ser informada de maneira segura. Na Figura 10.1, dois usuários precisam trocar uma informação sigilosa: o número do cartão de crédito. Neste caso, a mensagem (1) é embaralhada usando uma chave (2) com um algoritmo de criptografia (3). O resultado é um texto cifrado que, mesmo interceptado (4), não possibilita a identificação da mensagem original. Ao chegar ao destinatário, o texto cifrado é decriptado usando a mesma chave (5) e o mesmo algoritmo, resultando na mensagem original.

Figura 10.1 – Esquema de criptografia de chave simétrica

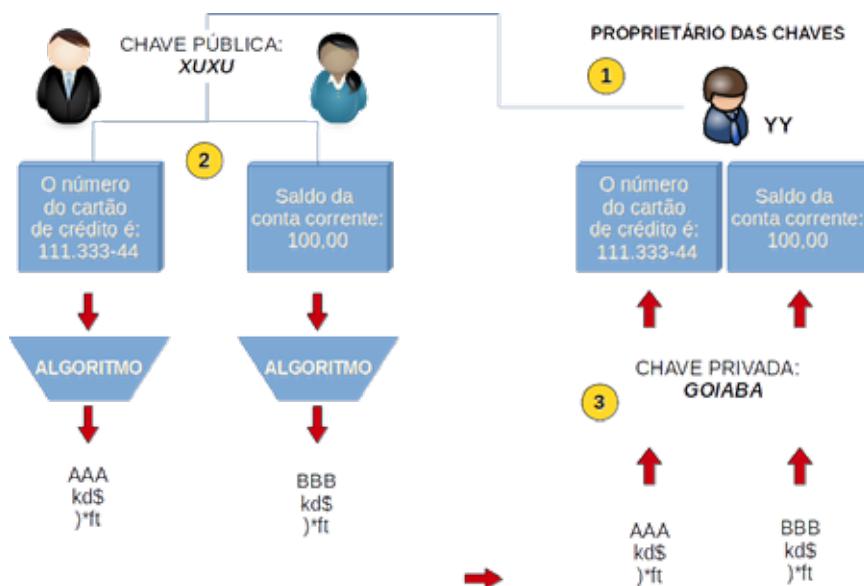


Fonte: Elaborada pelo autor.

A partir da década de 1970, foi criado um segundo tipo de criptografia: chave assimétrica. Nesse tipo, uma chave é utilizada para criptografar e outra chave é usada para descriptografar. Esse tipo também é conhecido como cri-

tografia de chave pública, pois uma das chaves (a pública) pode ser divulgada abertamente, já que não pode ser usada para descriptografar a mensagem de outros interlocutores. Um interlocutor gera as duas chaves: a chave pública pode ser distribuída para outros, mas a chave privada fica armazenada em um lugar seguro e não é compartilhada. Os interessados em enviar uma mensagem segura embaralham o texto usando a chave pública e enviam para o proprietário das chaves. Este, utilizando a chave privada, consegue reverter a mensagem para o texto original. Na Figura 10.2, o usuário YY (1) precisa receber as mensagens de maneira segura e para isso gera duas chaves: uma pública e uma privada. A chave pública (2) é divulgada abertamente e os demais usuários que precisarem enviar uma mensagem segura para o usuário YY a utilizam. Ao receber as mensagens criptografadas, o usuário YY utiliza a chave privada que ficou em seu poder para reverter a criptografia e acessar a mensagem original.

Figura 10.2 – Esquema da criptografia de chave assimétrica



Fonte: Elaborada pelo autor.

Ambos os tipos de criptografia são utilizados atualmente e cada um tem características que indicam em qual situação melhor se encaixa o algoritmo.

O fato de o tipo de criptografia de chave pública ter sido desenvolvido mais recentemente não indica que o de chave simétrica é menos seguro.

Todos os tipos de criptografia podem ser atacados pela força bruta: uma sequência de tentativas de várias combinações de chaves é testada até encontrar a que consegue reverter o texto cifrado. Por esse motivo, o que indica que um algoritmo de criptografia é considerado seguro é: 1) o custo da descoberta da chave é inviável ou, 2) o tempo que leva para a descoberta é muito grande. Em cada uma das situações, o esforço deve ser maior que o possível uso da informação. Periodicamente são reavaliados os algoritmos e o poder de processamento dos equipamentos para identificar se ainda é atendido pelo menos um dos dois requisitos.

Em muitos casos, os dois tipos de criptografia podem ser utilizados em conjunto, assim como os algoritmos podem ser usados repetidamente no mesmo texto cifrado para reforçar o embaralhamento. Uma questão importante associada aos tipos de criptografia e que impacta a utilização é o consumo de processamento: a criptografia de chave pública exige bem mais processamento que a criptografia de chave simétrica.

Os algoritmos de criptografia podem ser incorporados nos protocolos de comunicação para impedir a visualização de alguma informação que será utilizada para o acesso da rede ou na transmissão da informação.

## 10.2 Proteção do acesso

Quando é feito o compartilhamento de um recurso, é possível que ele esteja disponível para qualquer dispositivo ou usuário que tenha acesso à rede em que esse recurso esteja hospedado. Um mecanismo como o DNS pode facilitar a pesquisa do endereço IP e um protocolo como o HTTP permite que o conteúdo seja distribuído.

Mas quando o recurso não deve estar publicamente disponível e estiver liberado para apenas um conjunto restrito de pessoas, precisa ser implementada uma maneira de identificar a pessoa e depois avaliar se esta pode acessar o recurso.

Para avaliar as permissões e as restrições de acesso, são utilizadas tecnologias que implementam o triplo A (*AAA – Authentication, Authorization and*

*Accounting*, ou Autenticação, Autorização e Auditoria). A primeira controla a identificação do usuário, a segunda avalia que tipo de permissão esse usuário pode ter e a última determina qual tipo de rastreamento pode ser feito do consumo dos recursos feito pelo usuário. Essa última situação pode ser utilizada para a cobrança dos serviços.

O mecanismo de autenticação serve como primeira linha de defesa da rede e veremos tecnologias associadas a ele.

### 10.2.1 Autenticação

O processo de autenticação visa estabelecer um conjunto de mecanismos para comprovar a identidade do usuário (ou do serviço) que necessita de acesso a um recurso ou a uma rede. Para possibilitar essa comprovação, os mecanismos de autenticação precisam testar alguma condição que permita recusar tentativas de impostores. Para isso, devem se basear em atributos que estão relacionados a quatro meios gerais de comprovação da identidade:

1. algo que o usuário conhece (exemplo: senha);
2. algo que o usuário utiliza (exemplo: crachá);
3. algo que é próprio do indivíduo (exemplo: impressão digital);
4. algo que o usuário faz (exemplo: características da escrita manual).

Os dois últimos casos estão relacionados aos estudos de biometria: biometria estática e biometria dinâmica, respectivamente.

Conforme a tecnologia utilizada, caso seja comprovado por algum dos atributos citados, o identificador é considerado válido e pertencente à identidade original.

Várias dificuldades estão relacionadas aos meios de autenticação e devem ser consideradas na implementação: senhas podem ser interceptadas ou forjadas, crachás e tokens podem ser falsificados e as opções biométricas podem dar falsos positivos ou falsos negativos. Algumas opções têm custo maior que outras e podem influenciar na implantação, considerando o hardware necessário no leitor, a quantidade de leitores e (se não for uma solução biométrica) a quantidade de dispositivos a serem distribuídos para as pessoas. Em alguns casos, pode ser verificada a implementação de mais de um meio de auten-

ticação para o mesmo recurso, aumentando as etapas da autenticação, mas proporcionando uma segurança maior.

De qualquer maneira, as implementações dos mecanismos de autenticação são um importante aspecto da segurança das redes e estão incorporadas em pontos de entrada da rede, em protocolos de conexão, serviços em nuvem ou dispositivos. Podem evoluir de simples solicitação de uma palavra-chave até complexos protocolos com diferentes tipos de criptografia. Um dos mais utilizados mecanismos de autenticação é o Kerberos.

### 10.2.2 Kerberos

O Kerberos é um protocolo para autenticação e teve origem no projeto Athena do MIT (Massachusetts Institute of Technology), na década de 1980, para proporcionar um mecanismo de validação da identidade de um usuário sem que ele tenha de enviar as senhas pela rede. A versão 5, desenvolvida em 1989, tornou-se a versão padrão do Kerberos e foi disseminada para vários ambientes de rede, podendo utilizar a implementação *opensource* disponibilizada pelo MIT ou versões proprietárias.

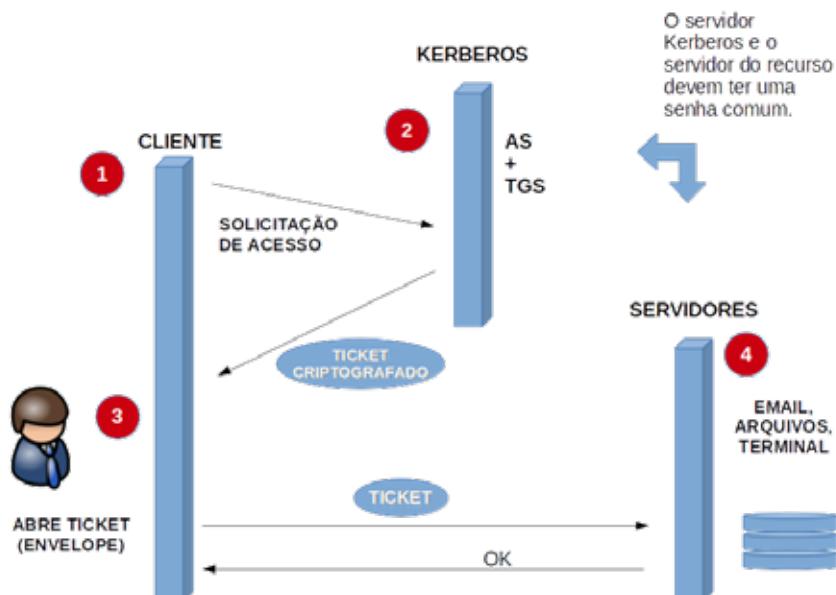
Esse sistema envolve o uso de criptografia simétrica e a troca de envelopes de dados (*tickets*) entre os servidores e vários algoritmos de criptografia podem ser utilizados. Durante o início da negociação para a autenticação, o usuário e o servidor Kerberos devem avaliar quais algoritmos de criptografia possuem em comum para dar continuidade ao processo.

#### Saiba mais

O sistema Active Directory (AD) da Microsoft baseia seu processo de autenticação da rede Windows no protocolo Kerberos. Utilizando um repositório central para o armazenamento dos usuários e das senhas, usa *tickets* para validar as identidades sem a transferência das senhas pela rede.

Na Figura 10.3, um exemplo do processo de autenticação usando o sistema Kerberos mostra o trâmite dos dados utilizados para validar a identidade do usuário sem que seja necessária a transmissão da senha em vários serviços da rede.

Figura 10.3 – Esquema de autenticação usando o Kerberos



Fonte: Elaborada pelo autor.

O usuário inicia o processo (1) e solicita ao servidor Kerberos o estabelecimento de um canal de confiança entre os dois. O serviço Kerberos encapsula uma chave temporária com outros dados de acesso a outros serviços em um envelope (2) e criptografa tudo usando como base a senha que tinha sido previamente registrada em sua base para aquele usuário. O usuário (3) abre o envelope usando a senha e retira o *ticket* que permitirá acessar o outro servidor. Esse ticket contém uma chave de acesso a outro servidor (4) que abrirá o *ticket* usando a senha que foi registrada no servidor Kerberos e, caso consiga descriptografar, considera o acesso validado.

Para cada serviço diferente é gerado um *ticket* pelo servidor Kerberos. Toda a cadeia de acesso fica validada, uma vez que o usuário consegue abrir o envelope com a senha, comprovando sua identidade, e o serviço de aplicação comprova sua identidade conseguindo também abrir seu envelope.



## Saiba mais

O protocolo LDAP (*Lightweight Directory Access Protocol*) é utilizado em vários sistemas para centralizar as informações sobre um grupo de objetos. Essa base é chamada de diretório e tem uma estrutura hierárquica (árvore) na qual objetos podem ser usuários ou programas. Para cada um dos objetos podem ser associadas informações que podem indicar uma política de acesso a um recurso (por exemplo, a permissão de execução de um programa) ou chaves de acesso criptografadas (por exemplo, a senha de um usuário). Pode ser utilizado em conjunto com o sistema Kerberos para centralizar as chaves (como é feito, por exemplo, no sistema AD nas redes Windows).



Os envelopes (*tickets*) têm um tempo de validade marcado pelo servidor Kerberos e devem ser destruídos ao encerrar a sessão. Por esse motivo, um protocolo de sincronia de relógios entre os dispositivos que fazem parte da rede deve ser implementado (ver protocolo NTP no Capítulo 6). Uma diferença no relógio entre dois dispositivos pode invalidar o *ticket* e impedir a autenticação.

### 10.3 Proteção da transmissão

Um dos aspectos ligados à segurança na transmissão de informações via rede é a possibilidade da interceptação dos dados durante a transferência das informações entre os dispositivos da rede. A única possibilidade de garantir alguma confidencialidade é com a utilização de criptografia nas informações.

Considerando que o uso da criptografia não é um processo trivial e nem sempre é identificado como necessário antes da transmissão das informações, alguns protocolos já implementam esse processo durante uma das fases da negociação para estabelecer a conexão. Também podem ser utilizados protocolos específicos para criar uma camada de segurança em conjunto com os outros protocolos da pilha TCP/IP.

No Quadro 10.1, vemos a identificação de alguns protocolos associados ao estabelecimento de uma comunicação criptografada entre dispositivos da rede e seu posicionamento na pilha TCP/IP. Os itens em cinza identificam a posição do protocolo em uma das camadas da pilha TCP/IP.

Quadro 10.1 – Tipos de segurança na pilha TCP/IP

Nível de rede			Nível de transporte			Nível de aplicação		
HTTP	FTP	SMTP	HTTP	FTP	SMTP	S/MIME		
TCP			SSL ou TLS			Kerberos	HTTP	SMTP
IP / IPsec			TCP			UDP	TCP	
			IP			IP		

Fonte: Stallings (2015).

Na camada de rede, o IPSEC é um dos protocolos desenvolvidos para atuar com os protocolos IPv4 e IPv6 e proporcionar serviço de autenticação e encapsulamento criptográfico para os protocolos das camadas superiores.

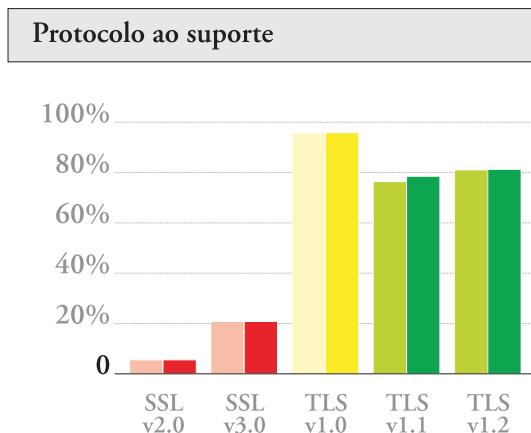
Com sua implementação, os protocolos de camadas acima da pilha TCP/IP utilizam a segurança proporcionada pelo IPSEC sem a necessidade de adaptação ou uso de portas diferentes para comunicação.

### 10.3.1 Secure Sockets Layer/Transport Layer Security

Em 1994, a empresa Netscape desenvolveu um protocolo chamado SSL (*Secure Sockets Layer*) para ser utilizado juntamente ao HTTP visando à proteção da transmissão das informações e assim a possibilidade do uso da internet em situações na quais a confidencialidade dos dados é imprescindível. Com isso surgiu a variação HTTPS (HTTP seguro) do protocolo de aplicação HTTP. As primeiras versões tiveram algumas vulnerabilidades detectadas até que o desenvolvimento da versão 3.0, em 1996, trouxe maior segurança para esse protocolo, sendo amplamente utilizado na internet. Em 1999, o IETF (*Internet Engineering Task Force*) desenvolveu um protocolo padrão de segurança baseado no SSL 3.0, chamado TLS (*Transport Layer Security*), e que vem evoluindo desde sua publicação. Atualmente, o TLS é a alternativa para a comunicação segura de vários tipos de aplicação, principalmente com relação ao protocolo HTTP, sendo o uso dos protocolos baseado no SSL considerado inseguro.

Na Figura 10.4, a proporção do uso de protocolos e suas versões utilizadas nos principais sites da internet.

Figura 10.4 – Suporte aos tipos de protocolos de segurança



Fonte: TrustWorthy Internet Movement (2016).

Existem vários tipos de navegadores com versões diferentes, assim como diferentes tipos de servidores que implementam protocolos de segurança. Ao estabelecer a negociação inicial, o navegador e o servidor avaliam quais são os protocolos disponíveis nos dois lados e, na lista avaliada, utilizam o mais recente.

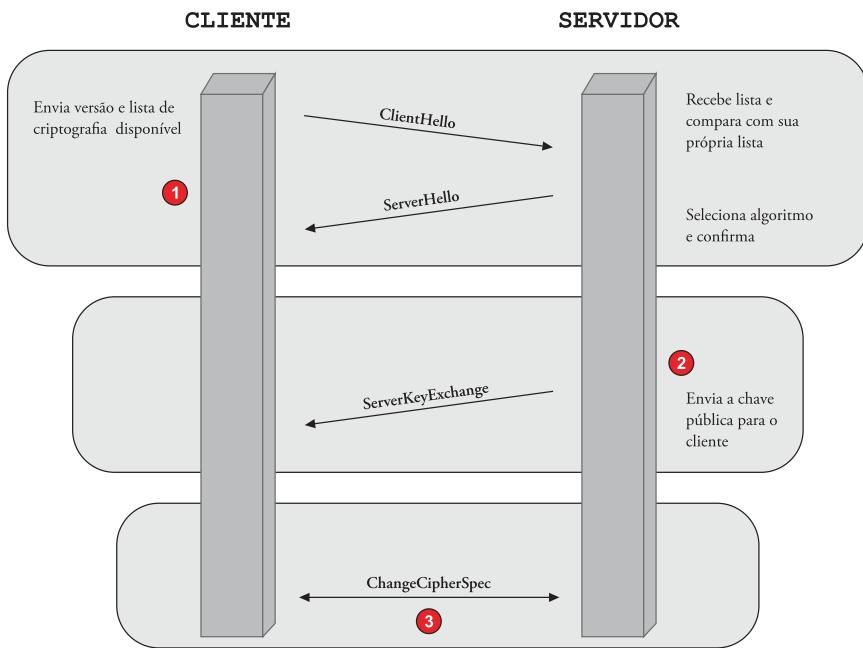
Esse processo inicial é chamado de *handshake* e é semelhante tanto para o SSL quanto para o TLS. Durante as negociações da versão dos protocolos a serem utilizados, como o TLS foi desenvolvido como uma evolução do SSL 3.0, os números de versão do TLS partem de 3.1 para o TLS 1.0, 3.2 para o TLS 1.1 e assim por diante.

No protocolo TLS, a transmissão das informações é criptografada usando uma chave e descriptografada do outro lado usando a mesma chave (ver criptografia simétrica em 10.1.1). Mas, como os dois sistemas (cliente e servidor) não se conhecem, a troca da chave deve ser feita de maneira segura para que um interceptador não a capture. Por esse motivo, o funcionamento do protocolo TLS envolve os dois tipos de criptografia. Usando uma chave pública, o cliente criptografa uma “chave secreta” e a

envia para o servidor. O servidor recupera essa “chave secreta” usando sua chave privada e a usa com um algoritmo de criptografia de chave simétrica em toda a transmissão.

Na Figura 10.5 pode ser verificado o processo inicial de negociação em que são tratadas as informações necessárias para o início da segurança da transmissão.

Figura 10.5 – Sequência de negociação SSL/TLS



Fonte: Elaborada pelo autor.

Na Figura 10.5, o cliente envia uma mensagem (1) de início da negociação (ClientHello) para o servidor. Dentro dessa mensagem estão informações sobre seus parâmetros, como os tipos e as versões dos protocolos de segurança e os tipos de criptografia que podem ser usados; o servidor, ao receber a mensagem, retorna com a escolha do protocolo e dos tipos de

criptografia a serem utilizados. Em uma mensagem seguinte (2), o servidor envia sua chave pública para o cliente. Ao receber a chave pública, o cliente cria uma chave internamente e criptografa usando a chave pública enviando para o servidor. A partir do recebimento, e conseguindo abrir a mensagem, o servidor pode usar a chave para criptografar toda a comunicação e descriptografar o retorno das mensagens do cliente. Nessa etapa (3), cliente e servidor enviam uma mensagem indicando o início da transmissão criptografada.

### Saiba mais

Um certificado externo contendo a chave pública do servidor também pode ser utilizado durante o processo de negociação (*handshake*) do TLS. Esse certificado é criado por um serviço externo (autoridade certificadora) e garante que o servidor com o qual o cliente está querendo se comunicar não é falso e portanto a chave pública pode ser usada para enviar a chave.

Durante o período de negociação, a transmissão não está criptografada e pode ser capturada. Mas como a chave é enviada para o servidor embalhada com a chave pública e apenas o servidor pode abrir a informação usando sua chave privada, o interceptador não pode acessar a chave para inspecionar o conteúdo.

#### 10.3.2 Navegação segura

O uso mais comum do SSL/TLS é com navegadores ao acessar algum sistema web que necessita de privacidade na comunicação. Na Figura 10.6 podem ser verificadas as informações utilizadas na conexão HTTPS de um site de e-mail. Ao clicar no cadeado que fica ao lado do endereço do site (figura que indica que a conexão é segura), o navegador apresenta uma janela com os detalhes da conexão. No fim da tela de detalhes, um botão <Export> permite a exportação do certificado.

Figura 10.6 – Visualização de certificado usado no navegador Firefox



Ao acionar a opção “about:config” na barra da URL, podem ser vistos os parâmetros utilizados no *handshake* do protocolo HTTPS. Ao habilitar esse acesso na opção de pesquisa (*search*), digitando a primeira parte do parâmetro “security:ssl3”, são listadas as opções relacionadas ao SSL v3 (podem ser verificadas as opções “security:ssl” e “security:tls”).

Figura 10.7 – Opção de configuração de variáveis de segurança no navegador Firefox

Preference Name	Status	Type	Value
security.ssl3_dhe_rsa_aes_128_sha	default	boolean	true
security.ssl3_dhe_rsa_aes_256_sha	default	boolean	true
security.ssl3_ecdh_ecdsa_aes_128_gcm_sha256	default	boolean	true
security.ssl3_ecdh_ecdsa_aes_128_sha	default	boolean	true

## 10.4 OpenSSL

O OpenSSL é um projeto *opensource* com o objetivo de disponibilizar os protocolos SSL e TLS para a comunidade. Esse sistema é utilizado por várias ferramentas que necessitam da camada de segurança proporcionada pelos dois protocolos.

Juntamente com a opção de utilizar esse sistema incorporado aos demais aplicativos, também é disponibilizado um utilitário em linha de comando que permite executar as principais atividades relacionadas à geração de chaves para a criptografia.

No Quadro 10.2, o comando *openssl* foi utilizado para verificar o conteúdo do certificado exportado pelo Firefox, salvo com o nome “mail.google.com.crt”. As mesmas informações podem ser vistas na janela <Visualização de certificado> da seção anterior.

Quadro 10.2 – Verificação de certificado usando comando *openssl*

```
# openssl x509 -text -in mail.google.com.crt

Certificate:
Data:
Version: 3 (0x2)
Serial Number: 2440643303400881353 (0x21dee82742a53cc9)
Signature Algorithm: sha256WithRSAEncryption
Issuer: C=US, O=Google Inc, CN=Google Internet Authority G2
Validity
    Not Before: Nov 10 15:56:49 2016 GMT
    Not After : Feb  2 15:31:00 2017 GMT
Subject: C=US, ST=California, L=Mountain View, O=Google Inc, CN=mail.google.com
Subject Public Key Info:
Public Key Algorithm: id-ecPublicKey
    Public-Key: (256 bit)
        pub:
            04:81:75:42:27:f7:8e:35:5a:73:93:6d:50:51:c9:
            21:78:8d:93:33:86:ed:67:b3:6b:15:28:78:f2:bc:
            de:98:25:c3:65:8c:b9:c9:72:de:77:60:83:19:fe:
            d9:99:7f:ce:0e:72:e4:6b:a0:b6:8c:7c:58:6f:85:
            5e:e5:42:db:fd
        ASN1 OID: prime256v1
[...]
```

Fonte: Elaborado pelo autor.

Com o comando *openssl* também é possível gerar uma chave privada e uma chave pública (ver Quadro 10.3).

Quadro 10.3 – Geração de chaves usando o *openssl*

```
# openssl genrsa -out chave-priv.pem 2048
# cat chave-priv.pem

-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAxK8uq/uad9F5y070xRcImlRDMSi7CTzNKO0y91rEBdFDhHEF
3180TXGMqw0bvXy9BpprZ1fzbPNQ/RNW2oAw6ivt2C7kc2+CqF0ZQIhxnlEddweF
mK4p91E+4AR0Htw3z7aw9taWr1ltdw2zdLS6n56UuiWcrxliIAVJQcp7zwaiia6
M/f00aygrhszh3Kw1ae4fitRLCamXIZJQWZQ4fqGkPclItF2zjg80VEktS6TkoKj
G325qmNRHbU4Gn+NOVzlxzPdDtyAL7Hais62obql4K0zBl7mtictwFr+B+fac
qub8yEn2nJ0qSpNLkIdKdpw0/Syy1Un+829fawIDAQABaIHAQCmdecvAVT8UbTtv
Lz4HOr1Ve59KebrhixTxTqgn7ZNVR1v/gunSPkZJrlleAfdfgammR4MqgB/C5a+uOc
auW4B4pehjNUctn8kecA1lqV3W8aQuj3GycXhgqxze1R92L381Vfx988epEz+28o
VJDNx06/U29mP+cj+w+0WAT4w3vpCfZ/2MIWjkOHjOEPiVXk146K+Wz8ekiu+2
UF1pTheuNYJ/Mpti2usAuk9RMf6dzn2hLNx6MaEdx8L54udPm5WQj7vH12hnqp8n
cWkyoYTE0QVLRhXKhsZXOnKq0pfmh7pViUM5PppF3XqOzzmwOrjJIUaSINjF
FHcaPMQBaogBAPFgoT+qva8hbAzPfmbPEzEWxwv+e8172scbywa/jln0aoZSl
q0TP+Ht1901GnTN2QsiPoDvmEQculPkoxko9U5FVuJwhjvn/h8X91N2FWAc9743e
2hNtkUVD6L6ocb0SwShpiJpeHUVFml38tVCr0U30/cyGOSuzKGxrAoGBANcZ
b0LMsDCPL6oZdTAXQwICV8fkA3/NK7zj50Ht53p69KUpCfJyPaR5pW0RVY6ML2d
bc9PN6Ecnr61bTOSEccwofOqmibqFQhcrRrwku818/H5EeEPSJpbhCWY/cJ8ml
Uxb0x2ZEvj6ut4WGscL+tr9WiYzzriNoaw5+ZkBaOGBALwWTBBMf9qDo6BUdls4
240ahh7mGbPFBG02QAI+E9uvppuY+o2NqgohLwo9ghYB/Eo1nVVNLW29Fcamo/
fh8y3HeXQk+So9MKwsApqd3Tiz4j8zswBbd4u9ypu033nA1nvOEpkEi2pb1qmK
JLq5i6U0xtgWv3MNW5U4e1EaoGAQtq8XbZDA3B0ktsz41v7WjvG70x82Tlwbw
19jbHIjdTGKwQ8x6ebydpRaUUp2BKJGGSAscu2cTOmoVPCawDSFmLmY18FJtgDE
c9RksFbRm1lorvIwhCPaMLaafpkBzgHDSwKD9NtIJZGNTwL1E00RkS8dhRoAy5L
ZTy3AgECgjAKhpEvylFgueubo1qKX7y/o/43yZait2+Tr07h8tg/z6Nx3iTiq8sif
Jas28AP0/qcJVLqBP1lODAiB1cEKyxGLT4B3HC69L/WyDAMS373Wp4SH4sKLmQ3WL
61TH0igrGoYDHSw1FR1xTsvjy+yMzj200EZS/oEY0jr9asKH3E49Uw==

-----END RSA PRIVATE KEY-----

# openssl rsa -in chave-priv.pem -out chave-pub.pem -pubout
# cat chave-pub.pem

-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIIBCgkCAQEAxK8uq/uad9F5y070xRcI
mlRDMSi7CTzNKO0y91rEBdFDhHEF3180TXGMqw0bvXy9BpprZ1fzbPNQ/RNW2oAw
6ivt2C7kc2+CqF0ZQIhxnlEddweFmK4p91E+4AR0Htw3z7aw9taWr1ltdw2zdLS
6n86UuiWcrxliIAVJQcp7zwaiia6M/f00aygrhszh3Kw1ae4fitRLCamXIZJQWZQ
4fqGkPclItF2zjg80VEktS6TkoKjG325qmNRHbU4Gn+NOVzlxzPdDtyAL7Hais6
2obql4K0zBl7mtictwFr+B+facqub8yEn2nJ0qSpNLkIdKdpw0/Syy1Un+829f
awIDAQAB
-----END PUBLIC KEY-----
```

Fonte: Elaborado pelo autor.

Além das chaves pública e privada, o *openssl* permite também a criptografia de arquivos usando uma chave simétrica e a geração de uma chave digital a partir de um arquivo (HASH).

### 10.4.1 Apache com SSL/TLS

No lado servidor, o domínio que precisa assegurar a privacidade da comunicação deve ser configurado para iniciar a negociação SSL/TLS. No Quadro 10.4 é exemplificada a configuração de ativação de SSL para um site. Apesar de a configuração envolver variáveis com a indicação de SSL no nome, elas servem também para o TLS. Na linha 2 está indicada a porta do protocolo TCP que é utilizada para a comunicação HTTPS.

Quadro 10.4 – Configuração SSL no servidor HTTP Apache

```
1. <IfModule mod_ssl.c>
2.   <VirtualHost *:443>
3.     ServerAdmin webmaster@localhost
4.     DocumentRoot /var/www/pagina_da_loja
5.
6.     SSLEngine on
7.
8.     SSLCACertificateFile /etc/apache2/ssl/loja.com.br.CA.crt
9.     SSLCertificateFile /etc/apache2/ssl/loja.com.br.pub.crt
10.    SSLCertificateKeyFile /etc/apache2/ssl/loja.com.br.key
11.
12.    ErrorLog ${APACHE_LOG_DIR}/loja_error.log
13.    CustomLog ${APACHE_LOG_DIR}/loja-access.log combined
14.
15.  </VirtualHost>
16. </IfModule>
```

Fonte: Elaborado pelo autor.

A linha 6 ativa o uso do módulo SSL no Apache. Nas linhas 8, 9 e 10 são indicados os certificados com as chaves privada (*SSLCertificateKeyFile*), pública (*SSLCertificateFile*) e a chave da entidade externa (*SSLCACertificateFile*) que pode validar se o servidor não é falso (não pertence ao domínio real).

#### Saiba mais

O servidor HTTP pode ser configurado para usar apenas a criptografia (SSL/TLS) sem uma certificadora externa. Nesse caso é feita a proteção da transmissão, mas não há a comprovação de que o site faz parte dos domínios de determinada empresa. Falsificadores podem configurar um servidor HTTP com SSL/TLS e alegar ser de uma loja conhecida. Por esse motivo, os navegadores solicitam confirmação ao encontrar um site protegido (SSL/TLS) mas sem um certificado comprovado.

## 10.5 Secure Shell

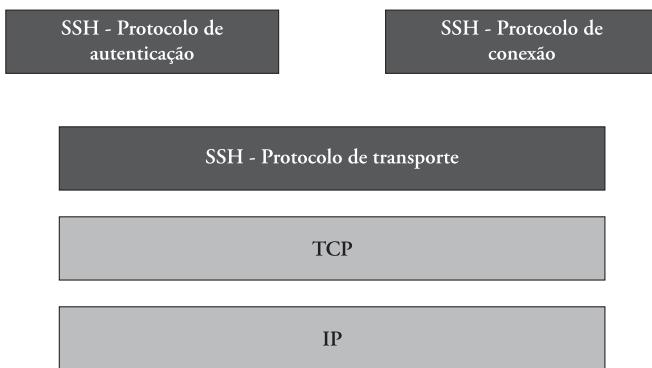
Ao administrar um servidor padrão Unix (por exemplo, Linux, FreeBSD, MacOS), por questões de praticidade, agilidade e desempenho, é comum a utilização da linha de comando para o diagnóstico e para atividades de administração do servidor e suas conexões. No início da internet, a segurança não era uma preocupação constante e, mesmo em redes locais, o uso do acesso remoto para a digitação dos comandos era feito com protocolos que transmitiam as informações de modo aberto (como vários protocolos vistos nos capítulos anteriores).

Um exemplo dessa situação era o protocolo da camada de aplicação chamado “*telnet*”, que permitia o acesso remoto à linha de comando do sistema operacional. A parte cliente que iniciava o acesso remoto ao servidor era iniciada a partir de um comando chamado também de “*telnet*”. Como todas as informações (usuário, senha e comandos) eram transmitidas abertamente pelo protocolo, surgiram vários incidentes de segurança que utilizavam as informações de autenticação interceptadas pelo protocolo para acesso não autorizado aos servidores. Em um desses incidentes, em 1995, Tatu Ylonen, da Universidade de Helsinki, na Finlândia, iniciou o desenvolvimento de uma alternativa para o acesso seguro aos servidores (SSH1). Após ser padronizado pelo IETF, a versão mais atualizada do protocolo (SSH2) passou a ser a versão recomendada.

Esse protocolo tem um mecanismo semelhante ao utilizado pelo SSL. É baseado na utilização de criptografia de chave pública para a autenticação inicial e depois no uso de criptografia de chave simétrica para a proteção da comunicação.

Essa negociação inicial é feita pelo serviço de autenticação do protocolo (Figura 10.8).

Figura 10.8 – Estrutura do protocolo SSH



Fonte: Elaborada pelo autor.

Uma implementação *open source* do protocolo é incluída em diversas versões de sistemas Unix como padrão. Em capítulos anteriores, o comando cliente “*ssh*” foi utilizado para exemplificar o acesso aos servidores remotos, inclusive no caso de servidor GNU/Linux em nuvem (Amazon EC2).

O protocolo SSH (*Secure Shell*) pode ser usado como um canal criptografado para outros protocolos. Por meio de um mecanismo de redirecionamento de porta, um utilitário que ativa um protocolo em uma porta específica pode usar a comunicação encriptada do SSH para encaminhar os pacotes e, no destino, o SSH volta a direcionar a comunicação para o serviço original.

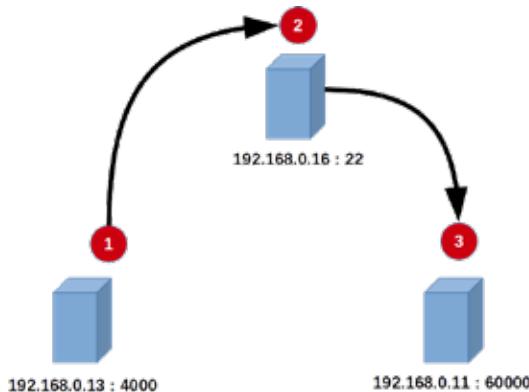
Com isso, apesar de o protocolo original não possuir mecanismo de segurança, o SSH cria um túnel pelo qual trafegam as informações de maneira segura. Esse mecanismo (chamado Port Forward – encaminhamento

## Rede de Computadores

de porta) pode ser usado em outros protocolos ou pode ser utilizado com o utilitário “*ssh*”.

No Quadro 10.5, há um modelo de encapsulamento de acesso a um servidor, usando um terceiro para fazer uma ponte. A única configuração para o túnel precisa ser feita no equipamento inicial. O sistema usado como ponte basta ter o servidor SSH ativo.

Quadro 10.5 – Esquema de túnel usando SSH



(servidor 3)

```
# nc -l 192.168.0.11 60000
```

(servidor 1)

```
# ssh -nNT -L 4000:192.168.0.11:60000 yanko@192.168.0.16  
yanko@192.168.0.16's password:
```

(servidor 1)

```
# nc localhost 4000  
oi tudo bem????
```

Fonte: Elaborado pelo autor.

No esquema anterior, é criado no servidor 3 um serviço de teste usando o utilitário “nc” (visto em capítulos anteriores), que abre uma conexão de espera na porta 60000.

No servidor 1 é usado o “*ssh*” com o parâmetro “-L” e a opção (4000:192.168.0.11:60000) indica que toda a conexão feita para a porta 4000 na máquina local será transferida para a máquina 192.168.0.11 na porta 60000 (servidor 3). Será usada a máquina 192.168.0.16 (servidor 2) com o usuário “*yanko*” como ponte para esse acesso (como indicado no fim do comando). A opção -nNT no início do comando é apenas para não abrir uma conexão Shell com o servidor de ponte.

Após o servidor 3 estar com a porta 60000 ativa e o comando “`ssh`” com o redirecionamento ligado, em outra janela no servidor 1 é feita uma conexão local para a porta 4000 (`nc localhost 4000`). Ao acessar a porta 4000, é redirecionado para o servidor 2, para a porta 22 (porta do SSH), por meio de uma conexão criptografada. O servidor 2 reencaminha a conexão para o servidor 3 para a porta 60000.

No servidor 1, ao acionar a captura de pacotes para verificar o acesso ao servidor 3 na porta 60000, não fica visível o acesso ao servidor 3. Na Figura 10.9 pode ser verificada a captura usando o aplicativo Wireshark.

**Figura 10.9** – Captura de tráfego no servidor 1

No.	Time	Source	Destination	Protoc	Length	Info
2	7.137687	192.168.0.13	192.168.0.16	SSH	118	Encrypted request packet len=32
3	7.225410	192.168.0.16	192.168.0.13	TCP	66	ssh > 37951 [ACK] Seq=2625915155 Ack=28595
4	9.069736	192.168.0.13	192.168.0.16	SSH	118	Encrypted request packet len=52
5	9.251354	192.168.0.13	192.168.0.16	SSH	118	[TCP Retransmission] Encrypted request seq=36
6	9.269406	192.168.0.16	192.168.0.13	TCP	66	ssh > 37951 [ACK] Seq=2625915155 Ack=28595
7	17.762021	192.168.0.13	192.168.0.16	SSH	118	Encrypted request packet len=52
8	17.763409	192.168.0.16	192.168.0.13	TCP	66	ssh > 37951 [ACK] Seq=2625915155 Ack=28595
9	19.481574	192.168.0.13	192.168.0.16	SSH	102	Encrypted request packet len=36
10	19.488877	192.168.0.16	192.168.0.13	TCP	66	ssh > 37951 [ACK] Seq=2625915155 Ack=28595
11	19.527848	192.168.0.16	192.168.0.13	SSH	138	Encrypted response packet len=72
12	19.527876	192.168.0.13	192.168.0.16	TCP	66	37951 > ssh [ACK] Seq=2859522294 Ack=26259
13	19.527216	192.168.0.13	192.168.0.16	SSH	102	Encrypted request packet len=36

No registro 2 da sessão do Wireshark, o acesso local à porta 4000 é redirecionado para o servidor 192.168.0.16 para a porta SSH (22). Com isso,

temos apenas a comunicação entre o servidor 1 (192.168.0.13) e o servidor 2 (192.168.0.16).

Na Figura 10.10, a captura de tráfego é realizada no servidor 2 (192.168.0.16). Nesse caso, no registro 3 da sessão, pode ser verificado o recebimento da conexão vinda do servidor 1 (192.168.0.13) para a porta SSH (22). Após a resposta confirmado o recebimento (ACK) no registro 4 da sessão, pode-se verificar a conexão entre o servidor 2 para a porta 60000 do servidor 3 (192.168.0.11).

Figura 10.10 – Captura de tráfego no servidor 2

No.	Time	Source	Destination	Protoc	Lengt!	Info
1	0.000000	192.168.0.13	255.255.255.25 DB-LSP-	197		Dropbox LAN sync Discovery Protocol
2	0.162991	192.168.0.13	192.168.0.255 DB-LSP-	197		Dropbox LAN sync Discovery Protocol
3	8.450142	192.168.0.13	192.168.0.16	SSH	118	Encrypted request packet len=52
4	8.480212	192.168.0.16	192.168.0.13	TCP	66	ssh > 37951 [ACK] Seq=2625918443 Ack=285952
5	8.480414	192.168.0.16	192.168.0.11	TCP	84	39615 > 60000 [PSH, ACK] Seq=2851483333 Ack=2851
6	8.514041	192.168.0.11	192.168.0.16	TCP	66	60000 > 39615 [ACK] Seq=3126294775 Ack=2851
7	11.967389	192.168.0.13	192.168.0.16	SSH	118	Encrypted request packet len=52
8	11.967430	192.168.0.16	192.168.0.13	TCP	66	ssh > 37951 [ACK] Seq=2625918443 Ack=285952
9	11.967556	192.168.0.16	192.168.0.11	TCP	75	39615 > 60000 [PSH, ACK] Seq=2851483351 Ack=2851
10	12.028855	192.168.0.11	192.168.0.16	TCP	66	60000 > 39615 [ACK] Seq=3126294775 Ack=2851

```

• Frame 3: 118 bytes on wire (944 bits), 118 bytes captured (944 bits)
• Ethernet II, Src: IntelCor_92:94:b5 (0c:d2:92:94:b5), Dst: HonHaiPr_9f:c2:6e (00:1e:4c:9f:c2:6e)
• Internet Protocol Version 4, Src: 192.168.0.13 (192.168.0.13), Dst: 192.168.0.16 (192.168.0.16)
• Transmission Control Protocol, Src Port: 37951 (37951), Dst Port: ssh (22), Seq: 2859527854, Ack: 26
• SSH Protocol

```

A cada acesso do servidor 1 ao servidor 2, na sequência inicia-se um acesso do servidor 2 ao servidor 3.

Com essas possibilidades, o protocolo SSH foi bastante flexível e permitiu a utilização desse sistema como uma opção genérica de criptografia para outros serviços, além do uso da linha de comando (Shell) remota. Em capítulos anteriores, foi verificado o uso do utilitário “*ssh*” para o acesso de linha de comando e o utilitário “*scp*” para realizar transferência de arquivos em canal criptografado.

Outra opção é a possibilidade de usar o SSH para manter uma conexão criptografada anexada diretamente ao sistema de arquivos. Semelhante ao uso do NFS visto no Capítulo 8.

## 10.6 Secure Shell File System

O SSHFS (Secure Shell File System) é um utilitário que combina a capacidade do protocolo FTP com a segurança do SSH e a facilidade de navegação do NFS que se associa diretamente na estrutura de arquivos.

No lado do servidor, basta ter o servidor SSH ativo. No lado cliente, o SSHFS cria uma conexão e a anexa ao sistema local de arquivos. A transferência de arquivos, a listagem e a verificação de conteúdo ficam transparentes para o usuário. Do mesmo modo que outros tipos de sistemas de arquivos anexados nos sistemas Unix, a conexão remota feita pelo SSHFS vira uma extensão da estrutura de pastas.

No Quadro 10.6, temos um exemplo de conexão e montagem de sistema local de arquivos usando o SSHFS. O servidor que possui o sistema de arquivos que será acessado (192.168.0.16) tem um servidor de SSH ativo.

Foi criada uma pasta local (`mkdir yyy`) como ponto de ligação entre a estrutura local de arquivos e a conexão remota. Após a execução do utilitário “`sshfs`”, os dois sistemas de arquivos (local e remoto) estão vinculados pela pasta `yyy`. Essa pasta pode ser usada para copiar, mover, apagar ou alterar arquivos, os quais fisicamente estão alocados no disco do servidor remoto (192.168.0.16) e toda a manipulação é feita por meio de um canal criptografado.

Quadro 10.6 – Conexão ao sistema de arquivos remoto usando o SSHFS

```
# mkdir yyy

# sshfs yanko@192.168.0.16: ./yyy
yanko@192.168.0.16's password:

# ls yyy
Área de Trabalho Documentos Downloads firefox Imagens livro.pdf
Modelos Música Público tj2015.pdf Videos
#
```

Fonte: Elaborado pelo autor.

No Quadro 10.7, o comando *mount* permite verificar quais sistemas de arquivos estão ativos localmente. Nesse caso, também é visualizada a conexão SSH com o servidor remoto e a pasta local a que está associada ( /home/yanko/yyy ).

Quadro 10.7 - Verificação e desmontagem da conexão SSHFS

```
# mount  
yanko@192.168.0.16: on /home/yanko/yyy type fuse.sshfs (rw,no-suid,nodev)  
# fusermount -u yyy
```

Fonte: Elaborado pelo autor.

No fim do quadro, o comando “*fusermount*” permite finalizar a conexão e liberar a pasta local do acesso remoto.

## 10.7 Privacidade on-line

Os protocolos utilizados na internet foram a base para a comunicação entre diversos pontos do Planeta, com a utilização de diversas mídias e diferentes tipos de linguagens e conteúdos. À medida que grande quantidade de dados sobre os usuários, seus conhecimentos e suas atividades foram sendo publicados, grupos e relacionamentos foram sendo criados e aumentaram ainda mais a complexidade da rede de informações. Também proporcionaram um ambiente em que essas informações e a relação entre os demais participantes pudessem municiar pessoas mal-intencionadas.

O uso de criptografia nos protocolos e nos serviços disponibilizados para o usuário está sendo difundido para que a comunicação possa ser mantida e ampliada. Mas a criptografia da comunicação é apenas um dos aspectos. O usuário deve também se preocupar com a segurança. Um estudo da EMC sobre a questão de privacidade on-line apontou, dentre outras questões, que 62% dos usuários não trocam senhas regularmente e 40% não alteram as configurações de privacidade em redes sociais.

## Síntese

Neste último capítulo, foram estudados os aspectos dos protocolos ligados a segurança, envolvendo o uso de criptografia para auxiliar a autenticação e a proteção da comunicação. Foi detalhado o mecanismo utilizado pelos protocolos SSL/TLS e SSH, usados como referência pelo capítulo. Foram verificados os exemplos da criptografia de chave pública e da criptografia de chave simétrica usados pelos protocolos. Um dos principais mecanismos de autenticação (Kerberos) foi usado como exemplo e verificado seu funcionamento.

## Atividades

1. Explique qual é a diferença entre os dois tipos de criptografia, considerando a utilização da chave usada para criptografar.
2. Marque verdadeiro (V) ou falso (F) com relação ao sistema de autenticação Kerberos.
  - ( ) O sistema Kerberos utiliza criptografia simétrica.
  - ( ) O sistema Kerberos utiliza uma chave pública para o início da autenticação.
  - ( ) O *ticket* que é enviado para o servidor contém um envelope criptografado com a senha e o usuário para o acesso aos recursos.
  - ( ) O Kerberos possui uma base com todos os usuários e suas senhas (chaves) no servidor central (AS + TGS).
3. Com relação ao protocolo SSL, pode ser utilizado pelos navegadores para criar uma comunicação segura, mas para isso os usuários devem utilizar uma senha diferente para cada acesso aos sites. O tamanho dessas senhas do usuário é garantir a segurança da comunicação.

Essa afirmação está: ( ) Correta ( ) Incorreta

Justifique.

4. Com relação ao protocolo SSH, marque verdadeiro (V) ou falso (F) nas questões:

- ( ) O SSH é utilizado apenas para digitação de comandos em terminal remoto.
- ( ) É utilizada apenas a criptografia simétrica na comunicação, pois ela é mais rápida.
- ( ) O SSH pode criar uma camada de criptografia para ser utilizada por outros protocolos.
- ( ) O conteúdo da transmissão dos dados via SSH pode ser consultado se usar um aplicativo como o Wireshark, que abre os pacotes da pilha TCP/IP.

# Conclusão

Ao chegar no final deste livro, percorremos uma série de assuntos voltados à área de redes de computadores, que foram pensados de maneira a ter uma boa compreensão do que envolve ser um administrador de redes.

É importante ressaltar que a área de redes não está isolada: ela interage com várias áreas da computação e atividades de negócio. O conhecimento de redes é importante para o profissional que precisa manter seu funcionamento, evoluir e criar alternativas. Mas é preciso compreender que ela é a base para que outras atividades (relacionadas à informática ou não) possam existir e evoluir.

O detalhamento de tecnologias de comunicação utilizadas em dispositivos, procurou apresentar sob qual base está construída a revolução digital em curso. Por meio de conjuntos de protocolos TCP/IP, a grande rede mundial chamada de Internet tem trazido novas formas de relacionamento entre pessoas, novos serviços e também desafios.

Entender como são trafegados os dados, de maneira que possam ser reconstituídos no outro dispositivo, foram os objetivos traçados nos capítulos.

A coleta da informação, a separação em partes menores para melhor compartilhamento do canal de comunicação, a organização e sequenciamento para permitir a reorganização dos dados, e a segurança do trajeto e do acesso. Estes processos devem estar claros para o profissional que deseja atuar na área de redes. Mas também são necessários para os demais profissionais de informática, uma vez que trabalhar isoladamente, sem acesso a outros profissionais ou recursos não cabe mais na rotina corporativa. E o conhecimento de como funcionam os protocolos e tecnologias de rede permite o entendimento mais abrangente do ambiente onde os sistemas e usuários estão inseridos.

Apesar de ter sido criado para o entendimento do assunto de redes, é importante não ficar limitado apenas ao conteúdo do livro. Ele é estático e o conhecimento vem evoluindo. Use o conteúdo como base para aprofundar o tópico que achar mais interessante. Os utilitários podem ter alternativas com mais opções sendo desenvolvidas e logo disponibilizadas para uso geral. Pesquise as referências e continue se atualizando.

Ao ler este livro, foi dado um grande passo. Continue caminhando.

# Gabarito

## 1. Cenário de Redes

1. Na LAN (rede local), como o termo dá a entender, os equipamentos estão na mesma localidade, portanto estamos falando de equipamentos no mesmo ambiente (mesma sala). Na MAN, temos uma abrangência envolvendo uma cidade, com interligação de equipamentos entre bairros. Por último, na WAN, temos a interligação de grandes distâncias (envolvendo países).
2. Podem ser citadas as vantagens de melhoria de desempenho por não compartilhar a capacidade do equipamento com os outros computadores e da segurança, pois as informações não serão replicadas para todas as portas do equipamento.
3. Os arquivos ficam armazenados no disco rígido dos servidores. Um servidor é um computador que possui um software que pode compartilhar seus recursos e suas informações na rede. O roteador é um equipamento específico de comunicação que atua na interligação de redes. O roteador encaminha o pacote de uma rede para outra.
4. O Access Point é um equipamento que permite a conexão de dispositivos mesmo em movimento e pode estabelecer uma conexão inclusive com obstáculos como paredes e armários. Com isso, um diretor com um notebook pode estar conectado ao sistema administrativo nos locais onde existe a abrangência do sinal, sem a necessidade de passar cabos e conectores antecipadamente por todos os locais nos quais poderiam ser feitas as reuniões.

## 2. Simulação de Redes

1. Opção (B). Conforme o texto, os endereços Ethernet não são variáveis (fixos em 48 bits), e é utilizada a base hexadecimal, não decimal, para indicar sua sequência. O broadcast é feito colocando todos os 48 bits em um (FF:FF:FF:FF:FF:FF).
2. O uso de VLAN nas portas do switch em que as máquinas de laboratório estão conectadas permite que elas fiquem isoladas do restante da rede da empresa.

3. O SSID permite identificar dentre as redes sem fio visualizadas pela placa wireless qual a que o equipamento vai estabelecer conexão. Como as ondas do sem fio atravessam objetos e paredes, se não houvesse uma maneira de identificar qual rede pertence a cada ambiente, os dispositivos não poderiam ficar próximos, pois as placas sem fio dos notebooks poderiam interligar nos access point de maneira aleatória.
4. O uso de simuladores pode auxiliar na reprodução de um problema, pois não necessita de investimento em hardware para criar o cenário a ser testado. Também permite avaliar alternativas a serem implementadas na rede real, além de poder testar as opções antes de executar as configurações nos aparelhos em uso na empresa.

### 3. Modelo OSI e Arquitetura TCP/IP

1. O protocolo é o Ethernet, usado no acesso ao meio de transporte físico (cobre, fibra óptica ou ar) através de uma interface de rede. Está relacionado à camada de link do TCP/IP, que é a camada que faz a interface entre o hardware e a camada superior.
2.
  - a) Antes era o IMP e atualmente é função do roteador.
  - b) Foi o esgotamento dos endereços IP de 32 bits.
  - c) Diminuir o risco de um conflito de nomes de domínios.
3. O protocolo SNMP pode ser ativado no switch para que as informações de tráfego possam ser coletadas remotamente. Com a quantidade de bytes entrando e saindo do switch, é possível ver a taxa de ocupação do equipamento.
4. O protocolo TCP faz o controle e pertence à camada de transporte.

### 4. Rede

1. Comentário: F – o protocolo DNS faz a conversão, o roteador trabalha apenas com endereço IP. V – 0 a 255 são a representação decimal

de 8 bits. F – os endereços privados foram faixas reservadas para uso em redes locais, não podendo ser vendidos ou utilizados diretamente na internet. V – o MX reflete o servidor de e-mail do domínio.

2. Com uma máscara de 23 bits para a parte da rede, sobram 9 bits do total de 32 bits do endereço. Com  $2^9$  bits temos um total de 512 endereços disponíveis.
3. O NAT possibilitou o aproveitamento das faixas de IP privados por várias empresas. Desta forma, em vez de desperdiçar um endereço público com um desktop ou impressora que não precisariam ser acessadas diretamente por algum outro equipamento na Internet, puderam ser utilizados os IPs da faixa privada. Como a faixa privada não é roteada na Internet, várias empresas podem usar o mesmo IP privado internamente sem a possibilidade de conflito com o IP privado da rede interna de outra empresa.
4. A máquina envia um pacote Ethernet com o endereço de *broadcast* como destino com os endereços de origem e destino do protocolo IP zerados. O servidor DHCP recebe o pacote e responde para o *mac address* um IP da lista de endereços disponíveis para aquela rede.

## 5. Camada de transporte

1. Usa-se a estrutura *socket* para criar uma identidade única para a conexão, envolvendo o IP de origem, a porta de origem, o IP de destino e a porta de destino.
2. Comentário: (F) As portas geradas aleatoriamente são as portas de origem da conexão. Os serviços precisam ter uma porta fixa para poderem ser encontrados; (V) Na confirmação, usa-se o número de sequência acrescido de 1 (ou o tamanho dos dados transmitidos); (V) Isso pode ser verificado nos testes do capítulo; (V) É usado o *flag FIN* ligado para finalizar a conexão TCP.
3. O protocolo ficaria esperando um longo tempo para o recebimento da confirmação que, inclusive, poderia não existir, caso o pacote tivesse sido perdido.

4. O protocolo ICMP é bloqueado em muitas redes e roteadores. Nesses casos, é necessário testar a conexão usando outros protocolos, como o TCP ou UDP, também usados na rede local.

## 6. Camada de Aplicação

1. O atraso da rede, ao transmitir o horário entre o servidor e o dispositivo local, deve ser avaliado ao calcular a diferença a ser ajustada no tempo usado no dispositivo local.
2. (V) A porta 80 é associada ao protocolo HTTP; (F) O código 200 indica que o comando foi recebido e entendido pelo servidor HTTP; (F) A URL pode ser utilizada para o acesso a um recurso na Internet e em outros serviços e utilitários, como visto nos comandos *ncat* e “*curl*”, por exemplo; (F) O CGI não é um protocolo e é utilizado para incorporar a execução de um programa no processo de resposta do servidor HTTP.
3. Os dois comandos retornam os parâmetros do HTTP sobre o conteúdo solicitado, mas o GET efetivamente transmite o conteúdo.
4. Permite o envio de opções e conteúdos fornecidos pelo usuário ao preencher o formulário. As informações são coletadas pelo navegador e enviadas por meio do HTTP para o servidor, em que podem ser utilizadas na montagem da resposta à solicitação.

## 7. Servidor HTTP e o Desenvolvimento WEB

1. (F) podem ser utilizadas portas de 0 a 65.535 no mesmo IP. A porta padrão não influencia na possibilidade de utilizar números de portas diferentes. (V) pode ser utilizado o mecanismo de CGI ou um módulo complementar com a linguagem de programação. (F) os módulos podem ser habilitados usando o comando “a2enmod” e não dependem do uso de JavaScript. (V) a variável “REMOTE\_ADDR” pode ser usada em um programa executado através da interface CGI para verificar o endereço IP do dispositivo cliente.

2. A variável contém informações sobre o sistema cliente que está fazendo a requisição. Se for identificado que foi feita a requisição de um dispositivo com sistema portátil “Android”, o servidor pode redirecionar o acesso para uma página específica.
3. Os itens podem sofrer manutenção (alteração, inserção ou remoção) e o formulário ficará atualizado sem intervenção manual no HTML.
4. O Ajax faz a requisição em paralelo com a operação da página e não necessita que a página seja totalmente reescrita para receber a resposta da solicitação.

## 8. Transferência e Compartilhamento de Arquivos

1. (V) o uso do sistema Samba permite o acesso via SMB aos compartilhamentos Windows dentro do Linux. (F) o Samba permite leitura, gravação e alteração, desde que o compartilhamento tiver liberado este tipo de acesso, apesar de serem sistemas de arquivos diferentes. (F) na linha de comando do Windows pode ser utilizado o comando *net* que faz acesso à rede SMB. (V) uma das características do SMB é permitir o acesso a impressoras e possibilitar a impressão de arquivos remotamente.
2. No Linux a estrutura de arquivos remota é anexada na estrutura local, funcionando como se estivesse navegando no sistema de arquivos. Podendo copiar, alterar ou listar os arquivos com os mesmos comandos usados no armazenamento local.
3. A SAN necessita de uma rede especial e acessa diretamente os blocos de bytes e a NAS utiliza a rede normal e acessa as informações através da estrutura de arquivos.
4. Os arquivos são verificados com relação a sua “idade”. Caso tenha diferença entre os arquivos entre a pasta de origem e a pasta de destino, os arquivos mais novos são transferidos para a pasta destino, mantendo as duas pastas com as mesmas informações atualizadas.

## 9. A Empresa e Seus Desafios

1. No serviço IaaS, todo o sistema operacional fica disponível para ser utilizado pela empresa, podendo instalar diversos tipos de ferramentas e aplicações. No SaaS, é utilizada uma aplicação pré-instalada e configurada para ser utilizada pela empresa, não sendo disponibilizada a opção para acesso ao sistema operacional.
2. A funcionalidade de *snapshot*, caso seja acionada antes da intervenção, pode retornar o estado do servidor para a condição anterior à manutenção.
3. Não há necessidade de investir em um ambiente com energia elétrica estabilizada ou refrigeração.
4. (a) **sim**: energia elétrica, refrigeração e conexões ficam sob responsabilidade do provedor; (b) **não**: o equipamento já faz parte da estrutura do serviço do provedor; (c) **não**: os dados podem ser armazenados em volumes virtuais ou *storages* virtuais (capítulo 8); (d) **não**: os sistemas tipo Windows também podem ser configurados na nuvem e acessados remotamente com TS (*terminal service*).

## 10. Segurança em Rede

1. A criptografia de chave simétrica utiliza apenas uma chave tanto para criptografar quanto para descriptografar. No mecanismo de chave pública, são utilizadas duas chaves: uma pública usada para criptografar as informações e uma privada para descriptografar.
2. (V) a criptografia é simétrica. (F) sendo a criptografia simétrica, não é utilizada chave pública. (F) o *ticket* não contém usuário e senha, e sim uma chave de acesso criada pelo servidor Kerberos. (V) para poder gerar as chaves de acesso e criptografar as mensagens usando a criptografia simétrica, o Kerberos precisa ter as senhas (chaves) antecipadamente, normalmente armazenada em uma base local.
3. Incorreta. O usuário não precisa interagir. O protocolo SSL tem um mecanismo para gerar automaticamente uma chave que será

trocada entre os sistemas por um canal criptografado usando a chave pública do servidor.

4. (F) o SSH pode ser usado para proteger outros protocolos, como e-mail ou FTP. (F) além da criptografia simétrica, é utilizada a criptografia de chave pública para a troca das chaves. (V) pode ser usado como um túnel criptografado para outros protocolos. (F) a criptografia impede a inspeção dos pacotes transmitidos pelo SSH por algum aplicativo que intercepte a comunicação.

# Referências

APACHE. **Apache HTTP Server Project.** Disponível em: <<http://httpd.apache.org>>. Acesso em: 30 out. 2016.

AWS – **Amazon Web Services.** 2016. Disponível em: <<https://aws.amazon.com/>>. Acesso em: 18 nov. 2016.

BERNERS-LEE, T. **Information management:** a proposal. Disponível em: <<https://www.w3.org/History/1989/proposal.html>>. Acesso em: 20 out. 2016.

BROWSER. **The WorldWideWeb browser.** Disponível em: <[https://www.w3.org/History/1994/WWW/Journals/CACM/screensnap2\\_24c.gif](https://www.w3.org/History/1994/WWW/Journals/CACM/screensnap2_24c.gif)>. Acesso em: 20 out. 2016.

CISCO. **Packet Tracer.** Disponível em: <[http://www.cisco.com/web/learning/netacad/course\\_catalog/PacketTracer.html](http://www.cisco.com/web/learning/netacad/course_catalog/PacketTracer.html)>. Acesso em: 8 set. 2016.

\_\_\_\_\_. **Software-Defined Networking:** Discover How to Save Money and Generate New Revenueplatform Overview. 2016. Disponível em: <<http://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/open-network-environment-service-providers/white-paper-c11-732672.pdf>>. Acesso em: 20 nov. 2016.

\_\_\_\_\_. **The Zettabyte Era – Trends and Analysis.** 2016. Disponível em: <<http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html>>. Acesso em: 28 ago. 2016.

COMER, D. E. **Interligação em rede com TCP/IP:** princípios, protocolos e arquitetura. Rio de Janeiro: Campus, 1998, v. 1.

COPPER. **Copper, cooper, everywhere:** which category cable should I choose?. 2016. Disponível em: <<https://www.copper.org/applications/telecomm/pdfs/a6125.pdf>>. Acesso em: 26 ago. 2016.

DARPA. **DARPA and the Internet Revolution.** Disponível em: <[http://www.darpa.mil/attachments/\(2015\)%20Global%20Nav%20-%20About%20Us%20-%20History%20-%20Resources%20-%2050th%20-%20Internet%20\(Approved\).pdf](http://www.darpa.mil/attachments/(2015)%20Global%20Nav%20-%20About%20Us%20-%20History%20-%20Resources%20-%2050th%20-%20Internet%20(Approved).pdf)>. Acesso em: 14 set. 2016.

DIERKS, T. et al. **The Transport Layer Security (TLS) Protocol**: Version 1.2, August 2008. Disponível em: <<https://tools.ietf.org/htmlrfc5246>>. Acesso em: 2 dez. 2016.

ECMASCRIPT. **ECMAScript® 2016 Language Specification**. Disponível em: <<http://www.ecma-international.org/ecma-262/7.0/index.html>>. Acesso em: 27 out. 2016.

EMC. **Global Internet Privacy Study Reveals Consumers' Conflicting Views**, 2014. Disponível em: <<https://uk.emc.com/about/news/press/2014/20140612-01.htm>>. Acesso em: 2 dez. 2016.

\_\_\_\_\_. **NFS**: Network File System Protocol Specification. Disponível em: <<http://brazil.emc.com/corporate/glossary/infrastructure-as-a-service.htm>>. Acesso em: 7 nov. 2016.

FACEBOOK. **HHVM**. Disponível em: <<https://code.facebook.com/projects/564433143613123/hhvm/>>. Acesso em: 27 out. 2016.

FREENAS. **Storage Operating System**. Disponível em: <<http://www.freenas.org>>. Acesso em: 11 nov. 2016.

GNS3. **What is GNS3**. Disponível em: <<https://gns3.com/software/>>. Acesso em: 10 set. 2016.

IANA. **Protocol registries**. Disponível em: <<http://www.iana.org/protocols>>. Acesso em: 9 out. 2016.

ICANN. **The Internet Corporation for Assigned Names and Numbers**. 2016. Disponível em: <<http://www.icann.org>>. Acesso em: 18 set. 2016.

IEEE. **IEEE Standard for Ethernet Amendment 2**: physical layer specifications and management parameters for 100 gb/s operation over backplanes and copper cables. 2016. Acesso em: 28 ago. 2016.

\_\_\_\_\_. **Welcome to the public listing for IEEE Standards Registration Authority**. Disponível em: <<https://regauth.standards.ieee.org/standards-ra-web/pub/view.html#registries>>. Acesso em: 4 set. 2016.

IETF TOOLS. **An architecture for describing**: Simple Network Management Protocol (SNMP) Management Frameworks. 2002. Disponível em: <<https://tools.ietf.org/html/rfc3411>>. Acesso em: 21 set. 2016.

\_\_\_\_\_. **The Interfaces Group MIB.** 2000. Disponível em: <<https://tools.ietf.org/html/rfc2863>>. Acesso em: 21 set. 2016.

IETF. **IANA Considerations and IETF protocol and documentation usage for IEEE 802 parameters.** Disponível em: <<https://tools.ietf.org/html/rfc7042>>. Acesso em: 5 set. 2016.

INTERNATIONAL TELECOMMUNICATION UNION. **Recommendation X.500 (11/88),** 2008. Disponível em: <<http://www.itu.int/rec/T-REC-X.500-198811-S>>. Acesso em: 1 dez. 2016.

IPV6. **IPv6.BR.** Disponível em: <<http://www.ipv6.br>>. Acesso em: 29 set. 2016.

\_\_\_\_\_. **Laboratório de IPv6.** Disponível em: <<http://ipv6.br/pagina/livro-ipv6/>>. Acesso em: 4 set. 2016.

ISO. International Organization for Standardization. **ISO/IEC 7498-1. 2015.** Disponível em: <<http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html>>. Acesso em: 14 set. 2016.

ITU. International Telecommunication Union. **ICT Statistics.** 2016. Disponível em: <<http://www.itu.int/en/ITU-D/Statistics/Pages/default.aspx>>. Acesso em: 29 ago. 2016.

JAVASCRIPT. **A short history of JavaScript.** Disponível em: <[https://www.w3.org/community/webed/wiki/A\\_Short\\_History\\_of\\_JavaScript](https://www.w3.org/community/webed/wiki/A_Short_History_of_JavaScript)>. Acesso em: 27 out. 2016.

JONES, M. T. **Sistemas de arquivos de rede e Linux.** Disponível em: <<https://www.ibm.com/developerworks/br/library/l-network-filesystems/>>. Acesso em: 12 nov. 2016.

KENT, S.; BBN CORP; ATKINSON, R. **Security Architecture for the Internet Protocol,** November 1998. Disponível em: <<https://tools.ietf.org/html/rfc2401>>. Acesso em: 2 dez. 2016.

KUROSE, J. F.; ROSS, K. W. **Rede de computadores e a internet:** uma nova abordagem. São Paulo: Addison Wesley, 2003.

LIBFUSE. **A network filesystem client to connect to SSH servers,** 2016. Disponível em: <<https://github.com/libfuse/sshfs>>. Acesso em: 1 dez. 2016.

- LYTLE, D. **Introduction to Storage Technologies SAN (Storage Area Networking) and FICON (FIber CONnection)**. Session 14274, August 12 2013. Disponível em: <<https://community.brocade.com/dtscp75322/attachments/dtscp75322/MainframeFICONsolutions/75/1/01+-+Session+14274+-+SAN+101+-+An+overview+of+Storage+Area+Networking+-+Presentation.pdf>>. Acesso em: 17 nov. 2016.
- MCKINLEY, H. L. **SSL and TLS: A Beginners Guide**, 2003. Disponível em: <<https://www.sans.org/reading-room/whitepapers/protocols/ssl-tls-beginners-guide-1029>>. Acesso em: 1 dez. 2016.
- MEIER, S. **IBM Systems Virtualization: Servers, Storage, and Software**. April 2008. Disponível em: <[www.redbooks.ibm.com/redpapers/pdfs/redp4396.pdf](http://www.redbooks.ibm.com/redpapers/pdfs/redp4396.pdf)>. Acesso em: 21 nov. 2016.
- MELL, P.; GRANCE, T. **The NIST Definition of Cloud Computing**. Gaithersburg, September 2011. Disponível em: <<https://dx.doi.org/10.6028/NIST.SP.800-145>>. Acesso em: 18 nov. 2016.
- MICROSOFT. **Microsoft Kerberos**, 2016. Disponível em: <[https://msdn.microsoft.com/en-us/library/windows/desktop/aa378747\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa378747(v=vs.85).aspx)>. Acesso em: 30 nov. 2016.
- MILLS, D. L. **A brief history of NTP time: confessions of an internet time-keeper**. Disponível em: <<http://www.eecis.udel.edu/~mills/database/papers/history.pdf>>. Acesso em: 20 out. 2016.
- MIT KERBEROS. **Kerberos: The Network Authentication Protocol**, 2016. Disponível em: <<https://web.mit.edu/kerberos>>. Acesso em: 1 dez. 2016.
- NET. **Net share**. Disponível em: <[https://technet.microsoft.com/pt-br/library/hh750728\(v=ws.10\).aspx](https://technet.microsoft.com/pt-br/library/hh750728(v=ws.10).aspx)>. Acesso em: 12 nov. 2016.
- NETCRAFT. **October 2016 Web Server Survey**. Disponível em: <<https://news.netcraft.com/archives/2016/10/21/october-2016-web-server-survey.html#more-23784>>. Acesso em: 29 out. 2016.
- NEUMAN, C. et al. **The Kerberos Network Authentication Service (V5)**, July 2005. Disponível em: <<https://tools.ietf.org/html/rfc4120>>. Acesso em: 2 dez. 2016.

**NIC. Núcleo de Informação e Coordenação do Ponto BR.** 2016. Disponível em: <<http://nic.br>>. Acesso em: 18 set. 2016.

**NIST. Time and frequency division.** Disponível em: <<https://www.nist.gov/pml/time-and-frequency-division/timekeeping-and-clocks-faqs>>. Acesso em: 22 out. 2016.

**NODEJS. About Node.js.** Disponível em: <<https://nodejs.org/en/about/>>. Acesso em: 27 out. 2016.

**NRO. Number Resource Organization. Regional Internet Registries.** 2016. Disponível em: <<https://www.nro.net/about-the-nro/regional-internet-registries>>. Acesso em: 17 set. 2016.

**NTOP. High-speed web-based traffic analysis and flow collection.** 2016. Disponível em: <<http://www.ntop.org/products/traffic-analysis/ntop>>. Acesso em: 19 set. 2016.

**NTP. The NTP FAQ and HOWTO.** Disponível em: <<http://www.ntp.org/ntpfaq/NTP-a-faq.htm>>. Acesso em: 22 out. 2016.

**NTPBR. Ntp.br.** Disponível em: <<http://ntp.br>>. Acesso em: 20 out. 2016.

**ODL – OpenDaylight. Platform Overview.** 2016. Disponível em: <<https://www.opendaylight.org/platform-overview>>. Acesso em: 21 nov. 2016.

**OPENNMS. The Network Management Platform.** 2016. Disponível em: <<http://www.opennms.org/en>>. Acesso em: 22 set. 2016.

**PHPNETCRAFT. PHP just grows & grows.** Disponível em: <<https://news.netcraft.com/archives/2013/01/31/php-just-grows-grows.html>>. Acesso em: 31 out. 2016.

**REGISTRO.BR.** 2016. Disponível em: <<http://registro.br>>. Acesso em: 18 set. 2016.

**RFC 959, The File Transfer Protocol (FTP).** Disponível em: <<https://www.ietf.org/html/rfc959>>. Acesso em: 8 nov. 2016.

**RFC 1002. Protocol standard for a NetBIOS service on a TCP/UDP transport:** detailed specifications. Disponível em: <<https://www.ietf.org/html/rfc1002>>. Acesso em: 10 nov. 2016.

RFC 1034. **Domain names** – concepts and facilities. Disponível em: <<https://www.ietf.org/rfc/rfc1034.txt>>. Acesso em: 1º out. 2016.

RFC 1094. **NFS**: Network File System Protocol Specification. Disponível em: <<https://www.ietf.org/html/rfc1094>>. Acesso em: 8 nov. 2016.

RFC 1738. **Uniform Resource Locators (URL)**. Disponível em: <<https://www.ietf.org/rfc/rfc1738.txt>>. Acesso em: 21 out. 2016.

RFC 2045. **Multipurpose Internet Mail Extensions (MIME) part one**: format of internet message bodies. Disponível em: <<https://www.ietf.org/rfc/rfc2045.txt>>. Acesso em: 20 out. 2016.

RFC 2046. **Multipurpose Internet Mail Extensions (MIME) part two**: media types. Disponível em: <<https://www.ietf.org/rfc/rfc2046.txt>>. Acesso em: 20 out. 2016.

RFC 2460. **Internet Protocol, Version 6 (IPv6) Specification**. Disponível em: <<https://www.ietf.org/rfc/rfc2460.txt>>. Acesso em: 2 out. 2016.

RFC 2616. **Hypertext Transfer Protocol – HTTP/1.1**. Disponível em: <<https://www.ietf.org/rfc/rfc2616.txt>>. Acesso em: 22 out. 2016.

RFC 3513. **Internet Protocol Version 6 (IPv6) Addressing Architecture**. Disponível em: <<https://www.ietf.org/rfc/rfc3513.txt>>. Acesso em: 2 out. 2016.

RFC 3875. **The Common Gateway Interface (CGI) Version 1.1**. Disponível em: <<https://www.ietf.org/html/rfc3875>>. Acesso em: 28 out. 2016.

RFC 3986. **Uniform Resource Identifier (URI): generic syntax**. Disponível em: <<https://www.ietf.org/rfc/rfc3986.txt>>. Acesso em: 21 out. 2016.

RFC 5661 **Network File System (NFS) Version 4 Minor Version 1 Protocol**. Disponível em: <<https://www.ietf.org/html/rfc5661>>. Acesso em: 11 nov. 2016.

RFC 6335. **Internet Assigned Numbers Authority (IANA)**. Procedures for the management of the service name and transport protocol port number registry. Disponível em: <<https://www.ietf.org/rfc/rfc6335.txt>>. Acesso em: 7 out. 2016.

RFC 6838. **Media type specifications and registration procedures.** Disponível em: <<https://www.ietf.org/rfc/rfc6838.txt>>. Acesso em: 20 out. 2016.

RFC 7540. **Hypertext transfer protocol version 2 (HTTP/2).** Disponível em: <<https://www.ietf.org/rfc/rfc7540.txt>>. Acesso em: 22 out. 2016.

RICCIARDI, F. **Kerberos Protocol Tutorial**, 27 nov. 2007. Disponível em: <<http://www.kerberos.org/software/tutorial.html>>. Acesso em: 2 dez. 2016.

RNP. Rede Nacional de pesquisa. **Redecomep**. 2016. Disponível em: <<https://www.rnp.br/servicos/conectividade/redecomep>>. Acesso em: 28 ago. 2016.

RODRIGUES, I. D. **Tributo ao padre-cientista Roberto Landell de Moura**. Memorial Landell de Moura, 2016. Disponível em: <[http://www.memoriallandelldemoura.com.br/landell\\_vida\\_obra.html](http://www.memoriallandelldemoura.com.br/landell_vida_obra.html)>. Acesso em: 28 ago. 2016.

SAMBA, **Samba: an introduction**. Disponível em: <[https://msdn.microsoft.com/pt-br/library/windows/desktop/aa365233\(v=vs.85\).aspx](https://msdn.microsoft.com/pt-br/library/windows/desktop/aa365233(v=vs.85).aspx)>. Acesso em: 8 nov. 2016.

SANDVINE. **Global Internet Phenomena Report**. 2016. Disponível em: <<https://www.sandvine.com/trends/global-internet-phenomena/>>. Acesso em: 27 ago. 2016.

SCHECHTER, G. **Visualizing Facebook's PHP Codebase**. Disponível em: <[https://www.facebook.com/note.php?note\\_id=10150187460703920](https://www.facebook.com/note.php?note_id=10150187460703920)>. Acesso em: 27 out. 2016.

SHIREY, R. **Internet Security Glossary, Version 2**, August 2007. Disponível em: <<https://tools.ietf.org/html/rfc4949>>. Acesso em: 2 dez. 2016.

SMB. **Microsoft SMB Protocol and CIFS Protocol Overview**. Disponível em: <<https://www.samba.org/samba/docs/SambaIntro.html>>. Acesso em: 8 nov. 2016.

SNIA. **Advancing Storage and Information Technology**. Disponível em: <[http://www.snia.org/tech\\_activities/standards](http://www.snia.org/tech_activities/standards)>. Acesso em: 7 nov. 2016.

SOUSA, L. B. de. **Redes de computadores**: Dados, voz e imagem. São Paulo: Érica, 1999.

SPICEWORKS. **Network monitoring software.** 2016. Disponível em: <<https://www.spiceworks.com/free-network-monitoring-management-software/>>. Acesso em: 19 set. 2016.

STALLINGS, W. **Criptografia e segurança em redes:** Princípios e práticas. 6. ed. São Paulo: Pearson, 2015.

STEVENS, W. R. **TCP/IP Illustrated:** the protocols. Indiana: Addison-Wesley, 1994. v. 1.

SYNERGY. **Amazon Dominates Public IaaS and Ahead in PaaS; IBM Leads in Private Cloud.** October 30, 2016. Disponível em: <<https://www.srgresearch.com/articles/amazon-dominates-public-iaas-paas-ibm-leads-managed-private-cloud>>. Acesso em: 19 nov. 2016.

TANENBAUM, A. S., BOS, H. **Sistemas Operacionais Modernos.** São Paulo: Pearson Education do Brasil, 2016

TANENBAUM, A. S.; WETHERALL, D. J. **Redes de computadores.** São Paulo: Pearson Prentice Hall, 2011.

THE APACHE SOFTWARE FOUNDATION. **Apache Module mod\_ssl,** 2016. Disponível em: <[https://httpd.apache.org/docs/current/mod/mod\\_ssl.html](https://httpd.apache.org/docs/current/mod/mod_ssl.html)>. Acesso em: 1 dez. 2016.

THE INTERNET PROTOCOL JOURNAL. **The Internet Protocol Journal,** 2016. Disponível em: <<http://protocoljournal.org>>. Acesso em: 2 dez. 2016.

TRUSTWORTHY INTERNET MOVEMENT. **SSL Pulse:** Survey of the SSL Implementation of the Most Popular Web Sites, December 04, 2016. Disponível em: <<https://www.trustworthyinternet.org/ssl-pulse>>. Acesso em: 30 nov. 2016.

UOL. **Netflix fatura R\$ 1,1 bi no Brasil e ultrapassa o SBT.** 2016. Disponível em: <<http://tvefamosos.uol.com.br/noticias/ooops/2016/01/11/netflix-fatura-r-11-bi-no-brasil-e-ultrapassa-o-sbt.htm>>. Acesso em: 27 ago. 2016.

VSFTPD. **VsFTPd.** Disponível em: <<https://security.appspot.com/vsftpd.html>>. Acesso em: 5 nov. 2016.

**WALDROP, M.** **Darpa and the Internet Revolution.** 50 Years of Bridging the Gap, 2015. Disponível em: <[http://www.darpa.mil/attachments/\(2015\)%20Global%20Nav%20-%20About%20Us%20-%20History%20-%20Resources%20-%2050th%20-%20Internet%20\(Approved\).pdf](http://www.darpa.mil/attachments/(2015)%20Global%20Nav%20-%20About%20Us%20-%20History%20-%20Resources%20-%2050th%20-%20Internet%20(Approved).pdf)>. Acesso em: 14 set. 2016.

**WASHINGTON.** **Netflix now accounts for almost 37 percent of our Internet traffic.** 2016. Disponível em: <<https://www.washingtonpost.com/news/the-switch/wp/2015/05/28/netflix-now-accounts-for-almost-37-percent-of-our-internet-traffic/>>. Acesso em: 27 ago. 2016.

**WIKIPEDIA.** **MediaWiki.** Disponível em: <<https://www.mediawiki.org/wiki/MediaWiki>>. Acesso em: 28 out. 2016.

**XEN PROJECT.** **History.** 2016. Disponível em: <<https://www.xenproject.org/about/history.html>>. Acesso em: 21 nov. 2016.

**XNS, Xerox Network Systems.** Disponível em: <[http://docwiki.cisco.com/wiki/Xerox\\_Network\\_Systems](http://docwiki.cisco.com/wiki/Xerox_Network_Systems)>. Acesso em: 12 nov. 2016.

**ZABBIX, The Enterprise-class Monitoring Solution for Everyone.** 2016. Disponível em: <<http://www.zabbix.com>>. Acesso em: 21 set. 2016.





A área de rede de computadores é dinâmica e sua capacidade tem evoluído a passos largos. Mas ao mesmo tempo, baseia-se em protocolos que foram concebidos, em alguns casos, há mais de duas décadas.

Prática e teoria devem andar juntas. Uma deve reforçar a outra. Por este motivo, esse livro de Redes de Computadores, traz para o leitor uma apresentação dos principais conceitos e teorias envolvendo a pilha TCP/IP, acompanhados de utilitários e ferramentas que permitem praticar, testar e simular o conhecimento descrito nos capítulos. O leitor poderá utilizar as dicas de ferramentas em cenários reais de empresas para auxiliar o entendimento do ambiente de rede corporativo e, inclusive, redes residenciais. O profissional de redes, ao ler os conceitos e teorias dos capítulos, vai poder associar com a sua prática diária.

Com a dependência cada vez maior de suas redes, as empresas estão precisando de profissionais que possam entender o cenário e as tecnologias que estão envolvidos em seus processos de negócio, para apresentar alternativas e melhorias para a sustentação da rede destas empresas.

