SENAI

SÉRIE TECNOLOGIA DA INFORMAÇÃO - *HARDWARE*

SERVIÇOS DE REDES

SENAI

Iniciativa da CNI - Confederação Nacional da Indústria

SÉRIE TECNOLOGIA DA INFORMAÇÃO - *HARDWARE*

SERVIÇOS DE REDES

SENAI

Iniciativa da CNI - Confederação Nacional da Indústria

SÉRIE TECNOLOGIA DA INFORMAÇÃO - *HARDWARE*

SERVIÇOS DE REDES

Esta publicação foi elaborada pela equipe do Núcleo de Educação a Distância do SENAI de Santa Catarina, com a coordenação do SENAI Departamento Nacional, para ser utilizada por todos os Departamentos Regionais do SENAI nos cursos presenciais e a distância.

SENAI Departamento Nacional
Unidade de Educação Profissional e Tecnológica – UNIEP

SENAI Departamento Regional de Santa Catarina
Núcleo de Educação – NED

FICHA CATALOGRÁFICA

S491g

Serviço Nacional de Aprendizagem Industrial. Departamento Nacional.
Gestão de pessoas / Serviço Nacional de Aprendizagem Industrial. Departamento Nacional, Serviço Nacional de Aprendizagem Industrial. Departamento Regional de Santa Catarina . Brasília : SENAI/DN, 2012.
142 p. il. (Série Segurança do Trabalho).

ISBN 978-85-7519-484-3

1. Gestão de Pessoas 2. Trabalho em Equipe I. Serviço Nacional de Aprendizagem Industrial. Departamento Regional de Santa Catarina II. Título III. Série

CDU:005.95

SENAI **Sede**

Serviço Nacional de Aprendizagem Industrial
Departamento Nacional

Setor Bancário Norte • Quadra 1 • Bloco C • Edifício Roberto Simonsen • 70040-903 • Brasília – DF • Tel.: (0xx61) 3317-9001 Fax: (0xx61) 3317-9190 • http://www.senai.br

Lista de ilustrações

Sumário

Introdução

1

Nesta unidade curricular, nós iremos tratar dos aspectos relacionados aos serviços de redes. Os serviços de redes são caraterizados por aplicações que trabalham no paradigma cliente-servidor, na maioria dos casos, e estes fornecem facilidades para que os clientes possam inter-operar entre aplicações e outros usuários. Estas operações podem ser trocas de informações, disponibilização de conteúdo, filtragem , segurança, entre outras características dos serviços de redes.

Para o bom profissional, é importante estar sempre preparado, tanto nas competências técnicas quanto nas relacionais, para poder atuar pró-ativamente.

Localize-se, na matriz curricular a seguir e confira as unidades curriculares e respectivas cargas horárias.

Técnico Redes de Computadores

MÓDULOS	DENOMINAÇÃO	UNIDADES CURRICULARES	CARGA HORÁRIA	CARGA HORÁRIA DO MÓDULO
Básico	Básico	• Eletroeletrônica Aplicada • Montagem e Manutenção de Computadores • Ferramentas para Documentação Técnica	60h 160h 120h	340h
Específico I	Ativos de Rede	• Cabeamento Estruturado • Arquitetura de Redes • Comutação de Rede Local • Interconexão de Redes PR • Gerenciamento e Monitoramento de Rede	108h 80h 120h 96h 60h	464h
Específico II	Servidores de Rede	• Servidores de Rede • Serviços de Rede • Serviços de Convergência • Segurança de Redes	120h 120h 60h 96h	396h

Quadro 1 - Matriz curricular
Fonte: SENAI DN

É hora de entrar no mundo dos serviços de redes e começar a trilhar os caminhos do conhecimento. Procure levar teoria e prática alinhados, contruindo o seu conhecimento e desenvolvimento profissional. Bons estudos!

Anotações:

Os Serviços de Redes

2

A Internet como conhecemos hoje só existe por um motivo: os serviços de rede. De nada adiantaria os complexos sistemas de comunicação de dados que temos hoje, se não existissem as informações para preencher estes canais de comunicação. Fibras ópticas, *links* via satélite, rádio comunicação, entre outros meios, conectam o mundo. Hoje, em fração de milésimos de segundo, é possível acessar informações digitalizadas em qualquer parte do planeta.

Ao final desse capítulo, você terá subsídios para:

a) entender os aspectos que compõem os serviços de redes;

b) entender os tipos de arquiteturas básicas dos serviços de redes.

Para começar os estudos sobre os serviços de redes, que tal fazer uma rápida viagem no tempo e ver um pouco sobre a história da Internet? Então, aperte os cintos e boa viagem!

2.1 A INTERNET

A história da Internet se confunde, em termos, com o surgimento dos serviços de rede. O primeiro *e-mail* trocado, ainda no ano de 1961, no MIT (*Massachusetts Institute of Technology*), era apenas um experimento de um pesquisador universitário em busca de uma nova forma de se comunicar. A primeira página acessada, no ano de 1989, foi apenas reflexo de um curioso chamado Tim Bernes Lee em disponibilizar documentos na rede para que seus colegas visualizassem o conteúdo de forma mais estruturada.

fotógrafo

Com apenas estes dois exemplos podemos traçar um paralelo de hoje com modernos sistemas de B2B (*Bussiness-to-Bussiness*) faturando milhões de dólares por dia em negócios via Internet. Ainda, há milhares de aplicações no mundo proporcionando níveis de tecnologia inimagináveis em outras épocas. Temos o abandono ou a troca de modelos de negócio consolidados por anos, como por exemplo, o envio de cartas, ofícios, documentos contábeis, documentos judiciais em geral, etc.

É fácil constatar o sucesso dos serviços de rede hoje em dia, quer ver só? Acompanhe o Casos e relatos a seguir.

CASOS E RELATOS

Cartão de crédito

Rogério, em início de namoro, convidou sua namorada para jantar fora. Caprichou na escolha do restaurante. Arrumou-se todo e, no horário marcado, se encontraram. O jantar foi perfeito, mas na hora de pagar a conta, Rogério lembrou que não havia retirado dinheiro no caixa eletrônico. Por um momento ficou preocupado em passar vergonha na frente da namorada, mas logo lembrou-se de que ele estava com seu cartão de crédito. Chamou o garçom, que trouxe até a mesa onde estavam, a maquininha de cartão sem-fio, e a conta foi paga, sem problemas.

Essa é uma prova concreta de como os serviços de rede estão disseminados. Hoje em dia, quando o cliente paga uma conta com cartão, por exemplo, ele não precisa saber, que aquela máquina contém um chip que está conectado à máquina via tecnologia GPRS (diretamente via satélite) e à rede da companhia de cartão de crédito, e que toda transação desta operação está vinculada a um serviço de rede que está administrando o cartão.

Tal grau de sofisticação é somente um pequeno exemplo de como os serviços de rede estão presentes na vida de cada um de nós. Eles afetaram até o relacionamento entre as pessoas.

SAIBA MAIS

As novas tecnologias têm beneficiado, inclusive, pessoas com algum tipo de deficiência. Saiba mais sobre essa nova realidade, fazendo uma busca em um *site* de pesquisa. Digite "tecnologias para deficientes" e confira. Você vai se surpreender com os resultados encontrados.

Mas, toda essa sofisticação e praticidade dependem dos componentes que são utilizados nos serviços de redes. Esse é o assunto do nosso próximo item!

2.2 OS COMPONENTES DO SERVIÇO DE REDES

Hoje, praticamente não há distâncias entre as pessoas. Desde um astronauta na estação espacial até os amigos batendo papo em redes de relacionameto, tudo isso é proporcionado pelos serviços de redes.

fotógrafo

fotógrafo

Basicamente, um serviço de rede é composto por pelo menos um componente dos citados a seguir:

a) **servidor de rede**: computador que realiza a função de disponibilizar o serviço para os usuários. Na maioria dos casos, este componente é caracterizado por computadores físicos com sistemas operacionais de redes, disponibilizando serviços. Atualmente, o servidor de rede é muito disseminado em alguns casos que envolvam sistemas virtuais;

b) **cliente**: computador que solicita o serviço ao servidor por meio da rede. Na maioria dos casos, é um usuário que efetua a solicitação por meio de uma interface via rede, como por exemplo, uma pessoa que solicita via *browser* uma determinada página da Internet, porém, há casos em que computadores são clientes de serviços;

c) **protocolo**: a forma com que o serviço de rede funciona, ou seja, as diretrizes de como a conexão e a comunicação entre o cliente e o servidor acontecerão. Um exemplo disso é o protocolo TCP/IP, que determina como os serviços de sua pilha funcionarão, dando as diretrizes de conectividade entre os serviços que ele suporta.

FIQUE ALERTA

Cuidado para não confundir os servidores de redes físicos em termos de *hardware* com os servidores de redes em termos de *software*. É muito comum utilizarmos expressões relativas aos servidores e, baseados no contexto, podemos entender se estamos falando do *harware* ou do *software*.

Em uma visão de arquitetura de computadores, os serviços de redes podem ser constituídos de três formas: *two-tier*, *three-tier* e *peer-to-peer*. Observe a arquitetura *two-tier* na figura a seguir. Nela, pode-se visualizar dois componentes: sendo o primeiro, o cliente, que está solicitando ao servidor determinada operação; e o segundo, o servidor, que irá fornecer ao cliente uma resposta à solicitação. Este é um cenário bem comum nos serviços de redes e pode ser encontrado em operações simples, como requisições de atualização de tempo (NTP).

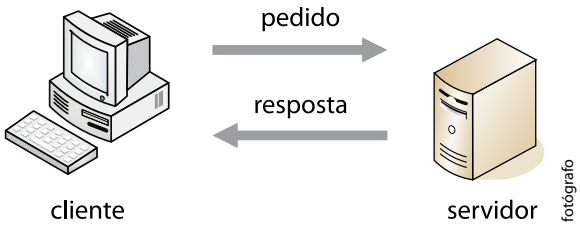


Figura 1 - Arquitetura *two-tier*

O segundo tipo de arquitetura para serviços de redes é a *three-tier*, mostrada na figura seguinte. Neste tipo de arquitetura, os clientes enviam as solicitações para um computador que conhece a infraestrutura e está totalmente integrado com outros servidores, desta forma, ele serve de interface para outros serviços. Neste caso, podemos citar os servidores de proxy, que intermediam solicitações dos clientes para fins de segurança da rede e ainda proporcionam um melhor desempenho da rede com os sistemas de *cache*.

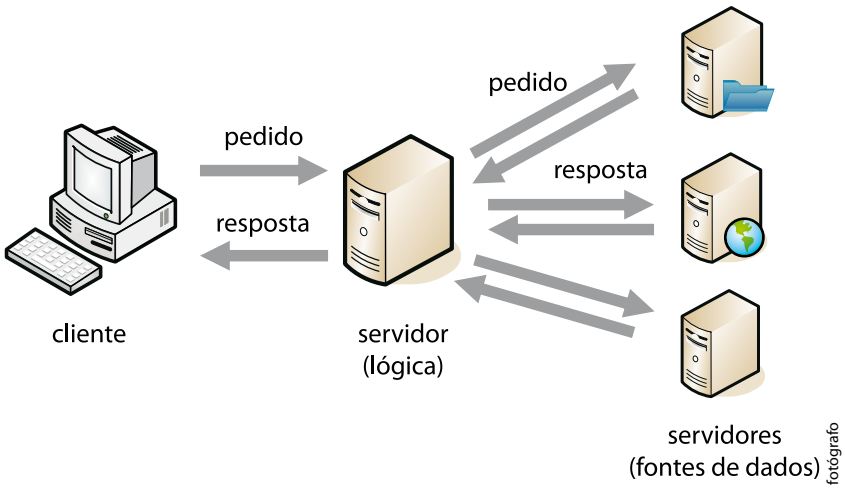


Figura 2 - Arquitetura *three-tier*

Por fim, o terceiro tipo de arquitetura de rede é a *peer-to-peer,* que você pode ver na próxima figura. Neste tipo de arquitetura, a caracterização é a colaboração entre os pares (peers). Neste tipo de ambiente há um compartilhamento constante de informações, com um processo normatizado baseado em protocolos populares.

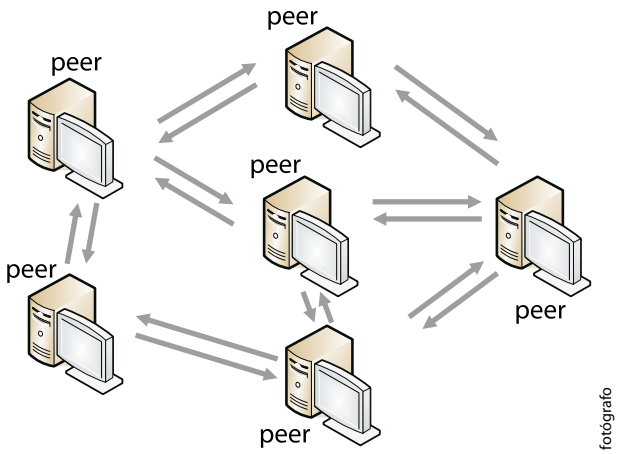


Figura 3 - Arquitetura *peer-to-peer*

VOCÊ SABIA?

Como exemplo deste tipo de arquitetura podemos citar as redes P2P, tais como: *e-Mule, Kazaa, Napster*, ou ainda, o novo protocolo Torrent, pois estas foram construídas sobre este paradigma para fins de troca de arquivos.

Há ainda outros paradigmas que utilizam este tipo de arquitetura, que são os Instant Messengers, tais como o MSN Messenger, ou ainda, o próprio Skype, sendo que ambos atuam na área de comunicação em redes *peer-to-peer* (P2P).

RECAPITULANDO

Nesse capítulo, você fez uma breve viagem na história da Internet, relembrando sobre o primeiro *e-mail* e primeira página acessada e conheceu, brevemente, alguns componentes que fazem parte dos serviços de rede. No próximo capítulo, você começará a desvendar este grande universo que é estruturar e manter serviços de redes, proporcionando aos usuários serviços de nome para identificar máquinas, e não endereços de IPs. Verá complexas estruturas de compartilhamento, trocas de informações, grandes sistemas de acesso e autenticação de serviços e saberá, também, como manter isso tudo atualizado, com todos os registros de operações. Aprenderá, ainda, como podemos guardar isso tudo de forma segura. Venha conosco nesta incrível jornada nos serviços de redes.

Servidor DNS

3

O serviço de nomes de domínio ou, mais conhecido como DNS (*Domain Name System*), é um importante componente da arquitetura dos servidores de rede. O seu entendimento, por parte do administrador de redes, é de vital importância para o funcionamento de outros serviços como *email*, Web e afins, pois na maioria dos casos, todos eles estão interligados. Neste capítulo, você aprenderá um pouco da história do DNS, entenderá o seu modelo de operação, conhecerá os registros e as mensagens dos servidores de nomes e verá, na prática, como se configura o serviço.

Ao final deste capítulo, você terá subsídios para:

a) conhecer a evolução e a história do DNS;

b) entender os modelos de operação do serviço DNS;

c) compreender a utilização dos registros e mensagens do serviço DNS;

d) entender o funcionamento do serviço DNS.

3.1 HISTÓRIA DO DNS

A Internet como conhecemos não existiria ou não teria a popularidade que tem hoje se não fosse pelo serviço conhecido como DNS (*Domain Name System*), acrônimo em inglês para Sistema de Nomes de Domínios. A sua importância é vital, pois com esta tecnologia é possível acessar endereços Web, enviar *e-mails*, acessar serviços, tudo isto baseado em nomes. O motivo básico para sua disseminação e popularidade é simples: para nós, seres humanos, é mais natural lembrarmos de nomes do que de números. Esta afirmação pode ser constatada no exemplo a seguir.

Se um amigo seu, chamado João, pedisse para você enviar um *e-mail* pra ele, com algumas informações relativas à matéria da semana passada, e lhe passasse seu endereço de *e-mail*: **joao@200.156.87.238**, seria mais fácil ou mais difícil de lembrar do que se ele pedisse para enviar um *e-mail* para **João@exemplo.com?**

Qual dos dois *e-mails* seria mais fácil de memorizar?

Outro exemplo é: seu amigo João lhe passa informações valiosas sobre determinada tecnologia que ele achou em um endereço na Web. Neste endereço existem informações que lhe interessam muito e que você gostaria de acessar posteriormente, para ler com mais calma e solidificar conceitos. Desta forma, você pergunta ao seu amigo em que *site* da Internet ele achou a informação. A reposta é rápida, seu amigo lhe fala que achou no *site* **http://102.4.89.253**. Seria mais fácil ou mais difícil que acessar o endereço **http://www.exemplo.com**?

A pergunta novamente é: qual dos dois endereços você memorizaria com mais facilidade?

Para ambos exemplos, é mais natural para nós lembrarmos de nomes e não de números. A facilidade que o DNS trouxe para a vida das pessoas é a principal razão de sua popularidade e ainda, foi a principal ferramenta na propagação da Internet. O modelo de operação utilizado, sua organização e estrutura tornaram o DNS um padrão na Internet.

Em termos técnicos, o serviço DNS é um esquema de gerenciamento de nomes, hierárquico e distribuído. Ele define a sintaxe dos nomes usados na Internet, as regras para delegação de autoridade na definição de nomes e estabelece um banco de dados distribuído, que associa nomes a atributos (entre eles os endereços IPs) e um algoritmo distribuído para mapear nomes em endereços.

O serviço de DNS pode ser encarado como uma base de dados distribuída, muito provavelmente, a base de dados que apresenta a maior abrangência geográfica e disseminação mundial. Hoje, a grande maioria dos serviços está integrada com esta tecnologia de endereços de *site*s a sistemas de comércio-eletrônico.

No início da Internet, quando ainda era chamada de ARPANET, a conversão entre o nome da máquina e o seu IP era realizada por meio de um arquivo denominado de *hosts.txt*. Os administradores de sistemas da época enviavam por *e-mail* as alterações dos seus domínios e buscavam via FTP *(File Transfer Protocol)* este arquivo, para atualizar seus sistemas. Veja, a seguir, o exemplo de um arquivo *hosts.txt*. Na parte esquerda, o endereço IP, e do lado direito, o nome a que este endereço IP está relacionado.

```
# Exemplo do arquivo hosts.txt
127.0.0.1   localhost
200.210.1.5  máquina1
4.67.91.233   máquina 2
```

Com o crescimento da Internet este mecanismo tornou-se inviável, visto que a quantidade de computadores entrando nesta nova rede aumentava exponencialmente a cada dia e a edição manual de um arquivo a cada nova alteração que ocorresse demandaria um trabalho significativo. Desta forma, com estas necessidades, surgiu o DNS, um sistema descentralizado, com as características necessárias para resolução dos problemas que eles tinham.

O DNS foi criado e desenvolvido no ano de 1983, por Paul Mockapetris, da Universidade do Sul da Califórnia (*University of Soul Califórnia* – USC), e representou um passo importante no desenvolvimento da Internet. Ele garantiu a divulgação e localização dos endereços dos recursos conectados à Internet. Este serviço ainda permitiu que informações sobre novos computadores e serviços fossem disseminados com maior agilidade, de acordo com a necessidade. O serviço DNS é especificado nas RFC *(Request for Comments)* 1034, 882, 883 e 973.

SAIBA MAIS

As Requisições por Comentários ou *Request for Comments*, comumente chamadas somente por RFC, são documentos que descrevem os padrões dos protocolos que definem a Internet. Estes documentos geralmente são lançados por pesquisadores, empresas ou organizações, como propostas de padrões para a Internet. Desta forma, pesquisadores de várias áreas de concentração no mundo avaliam a necessidade do padrão, sugerem melhorias, modificações, etc. Você encontra mais informações sobre este assunto em: <http://www.ietf.org/rfc.html>.

3.2 MODELO DE OPERAÇÃO

O modelo de operação do serviço DNS é constituído por um conjunto de servidores que mantém o banco de dados com os nomes e endereços das máquinas conectadas à Internet. Estes são distribuídos de forma hierárquica em vários locais do mundo, de modo que toda a operação não se concentre em um único ponto, o que sobrecarregaria o servidor e faria o processo de resolução de nomes lento e suscetível a falhas.

Os servidores DNS podem ser classificados em 3 (três) tipos: *root servers* (ou servidores raiz), *top-level domain* (TDL) e *autority name servers*. Conheça melhor cada um deles.

a) ***Root Servers* ou Servidores Raiz**: atualmente há treze *root servers* no mundo e eles são responsáveis por passar uma lista de nomes de domínios de alto nível correspondentes ao endereço IP que foi solicitado. Quando é falado que nós temos 13 servidores raiz no mundo, isto não quer dizer que sejam 13 servidores físicos, pois a totalidade deles tem até 10 servidores replicados com redundância. Atualmente, a arquitetura dos servidores de nomes raiz contém mais de 100 servidores físicos espalhados pelo mundo. (Root-Server, 2011).

b) ***Top-Level Domain* (TLD) ou Nomes de Domínio de Alto Nível**: estes são responsáveis pelos domínios denominados de alto nível. Exemplos deste tipo de domínios podem ser: com, org, net, uk, jp, br, edu, entre outros. Na maioria dos casos, estes TLDs são repassados para entidades governamentais ou privadas de cada país. Por exemplo: no Brasil, a CGI-BR (Comitê Gestor da Internet Brasileira) tem a autoridade sobre o TLD ".br". Isso significa que a CGI-BR tem autoridade sobre todos os domínios finalizados por ".br".

c) ***Authority Name Servers* ou Servidores de Nomes Autoritativos ou com Autoridade**: são o final do processo de resolução de nomes e comumente estão dentro das próprias organizações. Eles são responsáveis em repassar o endereço IP relativo ao nome que foi solicitado. Por exemplo: quando uma empresa adquire um nome com o final ".br", basicamente ela precisa registrar o domínio para o nome da empresa no *site* do CGI-BR.

VOCÊ SABIA?

Em 2011, o ICANN (*Internet Corporation for Assigned Names and Numbers* ou Corporação da Internet para Atribuição de Nomes e Números) autorizou a inserção de novos nomes de domínio à Internet. A partir de Janeiro de 2012, os registros de domínio poderão ser, por exemplo: "gmail.google" ou "reader.google". Não há os TLDs ".com, .net, .edu". O nome de domínio pode ser o próprio nome da empresa. Mais informações em: <http://www1.folha.uol.com.br/tec/932316-icann-aprova-novos-dominios-na-internet.shtml>.

De toda a arquitetura que compõem o serviço DNS há ainda os servidores DNS locais, que estão contidos dentro dos provedores. Estes servidores somente servem de consulta para os clientes do provedor e não fazem parte da hierarquia da arquitetura. A sua função é repassar o nome a ser resolvido para os servidores de DNS acima deles na arquitetura, servindo, basicamente, como um servidor de passagem. Observe a figura.

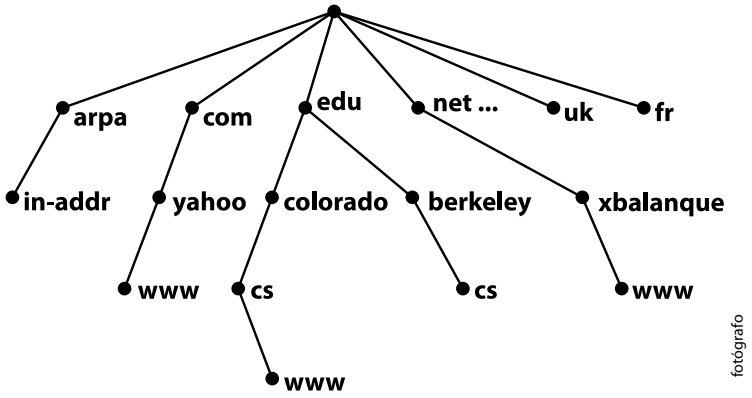


Figura 4 - Estrutura parcial e figurativa da Internet

Como você pôde ver, essa é uma figura ilustrativa da arquitetura hierárquica dos servidores DNS. Neste exemplo, para que um cliente acesse a URL (*Uniform Resource Locator* ou Localizador de Recurso Uniforme) "www.cs.colorado.edu", na Internet, primeiramente, o cliente envia uma requisição ao servidor DNS local do provedor, que envia uma requisição ao responsável pelo TLD ".edu" e este envia uma lista de servidores TLD que são responsáveis pelos domínios ".edu". Em seguida, o servidor de DNS local faz uma requisição a um dos servidores TLD buscando saber qual o endereço do "colorado.edu".

Este nome sendo resolvido, o servidor local envia uma requisição para o servidor DNS responsável pelo nome "colorado.edu" perguntando o endereço do servidor DNS "cs.colorado.edu". Por fim, o servidor DNS local envia uma requisição ao servidor responsável pelo nome "cs.colorado.edu" perguntando quem é "www.cs.colorado.edu" e obtém um endereço IP. Assim, o processo de resolução de nomes está finalizado.

Há um componente muito interessante nos servidores DNS que é o *cache*. Todo tipo de servidor DNS pode fazer uso de *cache* para dar mais desempenho ao servidor. O processo é o mesmo, porém quando um cliente solicitar, por exemplo, a URL <www.google.com>, o servidor efetuará todas as etapas vistas no exemplo da figura mostrada anteriormente, a fim de resolver o nome. Feito isto, durante um período de tempo, geralmente composto por 2 dias (isto pode ser configurável), o servidor não efetuará todas as etapas, e sim, entregará diretamente o endereço que está em seu *cache*, dando mais desempenho à solicitação.

3.3 REGISTROS E MENSAGENS

Você já sabe que o sistema DNS é regido por registros e mensagens. Devido a isto, os registros vistos na tabela a seguir figuram entre todos os sistemas operacionais, já que estamos falando de um padrão. Desta forma, o registro A (*address*) será visto em sistemas operacionais Linux, Windows e afins.

Existem muitos tipos de registros para muitos tipos de necessidades de *softwares*. Ainda, a cada dia novas propostas para novos registros estão sendo lançadas. É um processo contínuo desde o início da Internet. Na tabela a seguir, é possível visualizar os mais utilizados, com suas respectivas descrições e funções, porém, de forma alguma, ela cobre a totalidade de registros possíveis para o serviço DNS.

TIPO	DESCRIÇÃO	FUNÇÃO
A	*address record*	Retorna um endereço IP de 32-bits, IPv4, e é comumente utilizado para mapear nomes de máquinas (hostnames) a endereços IPs.
AAAA	*IPv6 address record*	Retorna um endereço IP de 128-bits, IPv6, e é comumente usado para mapear nomes de máquinas (hostnames) a endereços IPs.
CNAME	*Canonical name record*	Apelido de um nome para outro. É muito utilizado quando se quer determinar mais de um nome para a mesma máquina, mantendo somente um endereço IP.
MX	*mail exchange record*	Mapeia um nome de domínio para um Mail Transfer Agent.
NS	*name server record*	Delega uma zona DNS para ser usada em servidores de nomes autoritativos.
PTR	*pointer record*	O PTR geralmente é utilizado para DNS reverso, quando um cliente deseja saber o nome de um determinado IP.
SOA	*start of authority record*	Especifica informações autoritativas sobre zonas DNS, incluindo o servidor de nomes primário, o e-mail do administrador, o número serial do domínio e outras informações relacionadas a tempo.

Quadro 2 - Alguns tipos de Registros de DNS

Você conferiu informações sobre os registros e mensagens. Agora, confira como é feita a instalação e configuração de um servidor DNS.

3.4 INSTALANDO E CONFIGURANDO UM SERVIDOR DNS

Então, preparado para instalar e configurar um servidor DNS? Para aprender, vamos instalar e configurar um servidor de nomes de domínios em um sistema operacional *Debian GNU/Linux Squeeze*. O *software* escolhido para a estruturação do servidor é o BIND da *Internet Systems Consortium* – ISC. Este é o servidor de nomes mais utilizado na Internet e também é o *software* que está presente nos Servidores de Nomes Raiz da Internet. As operações aqui vistas podem ser aplicadas em qualquer sistema operacional Linux, visto que apenas o sistema gerenciador de pacotes mudará. O processo de instalação pode ser visto a seguir.

```
root@server:/ # apt-get install bind9
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
bind9utils
Suggested packages:
bind9-doc resolvconf ufw
The following NEW packages will be installed:
bind9 bind9utils
0 upgraded, 2 newly installed, 0 to remove and 11 not
upgraded.
Need to get 476 kB of archives.
After this operation, 1,290 kB of additional disk spa-
ce will be used.
Do you want to continue [Y/n]? y
Get:1 http://security.debian.org/ squeeze/updates/main
bind9utils amd64 1:9.7.3.dfsg-1~squeeze3 [121 kB]
Get:2 http://security.debian.org/ squeeze/updates/main
bind9 amd64 1:9.7.3.dfsg-1~squeeze3 [355 kB]
Fetched 476 kB in 0s (928 kB/s)
Preconfiguring packages ...
Selecting previously deselected package bind9utils.
(Reading database ... 195078 files and directories cur-
rently installed.)
Unpacking bind9utils (from .../
bind9utils_1%3a9.7.3.dfsg-1~squeeze3_amd64.deb) ...
Selecting previously deselected package bind9.
Unpacking bind9 (from .../bind9_1%3a9.7.3.dfsg-
-1~squeeze3_amd64.deb) ...
Processing triggers for man-db ...
```

```
Setting up bind9utils (1:9.7.3.dfsg-1~squeeze3) ...
Setting up bind9 (1:9.7.3.dfsg-1~squeeze3) ...
Adding group `bind' (GID 126) ...
Done.
Adding system user `bind' (UID 117) ...
Adding new user `bind' (UID 117) with group `bind' ...
Not creating home directory `/var/cache/bind'.
wrote key file "/etc/bind/rndc.key"
# Starting domain name service...: bind9.
```

A instalação pode ser feita com o commando "apt-get" diretamente na linha de comando do sistema operacional. Nos códigos apresentados, você pôde visualizar a instalação do pacote. No processo, as dependências são resolvidas e é solicitada uma confirmação da operação. Após a confirmação, inicia-se o download dos pacotes e, após isto, as dependências são instaladas, bem como o *software* em si. Ainda é criado um grupo e um usuário no sistema operacional, estes dois serão utilizados para executar o *software*.

Após a instalação, é criada uma pasta no sistema operacional, no diretório "/etc". Esta pasta contém os arquivos de configuração do servidor DNS. Veja a seguir, o conteúdo deste diretório.

```
root@molar:/# ls -l /etc/bind
total 52
-rw-r--r-- 1 root root  271 Jul  5 13:52 db.127
-rw-r--r-- 1 root root  353 Jul  5 13:52 db.empty
-rw-r--r-- 1 root root  270 Jul  5 13:52 db.local
-rw-r--r-- 1 root root 2994 Jul  5 13:52 db.root
-rw-r--r-- 1 root bind  463 Jul  5 13:52 named.conf
-rw-r--r-- 1 root bind  165 Jul  5 13:52 named.conf.
local
-rw-r--r-- 1 root bind  572 Jul  5 13:52 named.conf.
options
-rw-r----- 1 bind bind   77 Sep 12 15:28 rndc.key
-rw-r--r-- 1 root root 1317 Jul  5 13:52 zones.rfc1918
```

Confira a explicação do que é cada um dos itens relatados no conteúdo do diretório.

a) **db.127**: mapa da zona reversa do *localhost*. É uma zona padrão dos servidores Bind9. Quando um cliente perguntar qual o nome do endereço 127.0.0.1, ele responderá que é o *localhost*.

b) **db.empty**: mapa vazio, padrão no servidor Bind9.

c) **db.local**: mapa de resolução de nomes para o *localhost*. É uma zona padrão dos servidores Bind9. Serve para que, toda vez que algum cliente perguntar qual o endereço do nome *localhost*, ele responderá que é 127.0.0.1.

d) **db.root**: mapa da zona "hint". Neste mapa ficam os endereços dos servidores raiz (*root-servers*). É uma zona padrão dos servidores de DNS em geral, e nela estão listados os endereços dos servidores raiz na Internet.

e) **named.conf**: arquivo de configuração principal do sistema de DNS. No sistema operacional Debian, ela foi segmentada em 3 arquivos, sendo eles: named.conf, named.conf.local e named.conf.options. O primeiro deles contém somente as zonas padrão; o segundo é usado para que novas zonas sejam armazenadas; e o terceiro é para configurações de segurança e desempenho. Em outras distribuições, algumas vezes, está tudo contido dentro do named.conf.

f) **named.conf.local**: arquivo para inserção de novas zonas.

g) **named.conf.options**: arquivo para inserção de novos parâmentros de desempenho e segurança.

h) **rndc.key**: arquivo que armazena a chave de compartilhamento de mapas. Isto é opcional.

No arquivo de configuração a seguir, é possível visualizar o conteúdo do "named.conf" do Bind.

```
root@server:/# cat /etc/bind/named.conf

zone "." {
      type hint;
      file "/etc/bind/db.root";
};

zone "localhost" {
      type master;
      file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
      type master;
      file "/etc/bind/db.127";
};

zone "exemplo.com" {
      type master;
      file "/etc/bind/db.exemplo.com";
};

zone "1.168.192.in-addr.arpa" {
      type master;
      file "/etc/bind/db.exemplo.rev";
};
```

Nessa cofiguração, é possível visualizar as zonas: *hint*, *localhost*, a zona reversa do *localhost* (127.in-addr-arpa) exemplo.com, e sua zona reversa (1.168.192.in-addr.arpa). Estas duas últimas foram criadas por fator de exemplo e não vêm por padrão na instalação do Bind9.

FIQUE ALERTA

Quando procuramos saber o nome de um determinado IP é importante prestar atenção na notação "1.168.192.in-addr.arpa". Neste caso, estamos trabalhando com uma rede classe C, com máscara 255.255.255.0. Desta forma, devemos inserir "1.168.192" (lê-se da direita pra esquerda) antes da notação padrão ".in-addr.arpa". Se nós, por exemplo, estivéssemos trabalhando com uma classe B com máscara 255.255.0.0, deveríamos inserir somente "168.192" antes da notação padrão ".in-addr.arpa", porém, dentro do mapa, deveríamos colocar os dois últimos octetos do endereço IP.

A zona "hint" é a principal zona de um servidor DNS. Ela é reponsável por apontar os Servidores de Nomes Raiz que estão localizados no arquivo de configuração "db.root" na mesma pasta do named.conf. Se esta zona fosse apagada, o serviço não conseguiria resolver nomes externos que estivessem fora de seu cache porque o sistema não saberia a quem consultar por novos TLDs.

Seguindo a linha de nosso exemplo, toda zona necessita de um mapa. Este mapa é figurado em um arquivo de texto que está escrito em um formato definido pelo servidor e nele estão contidas as informações (registros, endereços, nomes, etc.) sobre o domínio. Na configuração apresentada a seguir, pode-se visualizar um exemplo de um mapa para a zona "exemplo.com".

```
$TTL  604800
@     IN    SOA   ns.exemplo.com. root. exemplo.com. (

 2011110911       ; Serial

 604800           ; Refresh
                    86400          ; Retry

      2419200           ; Expire

 604800 )          ; Negative Cache TTL
;
@     IN    NS    ns.exemplo.com.
@     IN    A     192.168.1.1
;
@     IN    MX 5  mail.exemplo.com.
;
ns    IN    A     192.168.1.1
mail  IN    A     192.168.1.2
www   IN    A     192.168.1.3
```

Como você pôde ver, as informações sobre o domínio estão figuradas em um mapa que está relacionado à zona "exemplo.com". Veja, a seguir, a explicação de cada uma das linhas.

a) **TTL**: é o registro do tempo de vida do mapa, medida em segundos. No caso, o TTL 604800 representa uma semana ou sete dias.

b) **SOA**: define informações autoritativas da zona, entre elas: servidor de nomes, administrador da zona, registro serial, entre outras informações.

d) **NS**: define o servidor de nomes da zona.

e) **MX**: define o servidor de *e-mails* do domínio.

f) **A**: define a relação entre o nome da máquina e o endereço IP relacionado.

O registro SOA contém, entre outras coisas, a informação do serial do mapa, composto, geralmente, por 10 dígitos e é muito importante para o sistema DNS. Todas as informações sobre versões de mapas são propagadas baseadas nele. Por exemplo: se nosso número serial é 2011091101 e eu faço uma alteração inserindo um novo registro e não troco o número serial, esta modificação não será propagada, ou seja, ninguém conseguirá resolver o nome.

FIQUE ALERTA

O símbolo @ é responsável por vincular a informação ao nome da zona. Lê-se o @ sempre como o nome do domínio. No quadro dos comandos, ele representa o domínio "exemplo.com".

Os comandos a seguir mostram o mapa de resolução reversa do domínio "exemplo.com". O leitor deve ficar atento porque o servidor DNS não só resolve nomes para endereços, mas também, resolve endereços para nomes. Por exemplo: um cliente pode solicitar ao servidor DNS qual o nome de um determinado endereço IP. Isto é muito comum e bastante utilizado, por exemplo, no combate a SPAMs. Veja:

```
$TTL  604800
@     IN    SOA   ns.exemplo.com. root.exemplo.com. (

      2011110901        ; Serial

 604800           ; Refresh
                    86400          ; Retry

      2419200           ; Expire

 604800 )   ; Negative Cache TTL
;
@     IN    NS    ns.exemplo.com.
1     IN    PTR   ns.exemplo.com.
2     IN    PTR   mail.exemplo.com.
3     IN    PTR   www.exemplo.com.
```

A configuração que você acabou de ver representa o mapa de resolução reversa de nomes do domínio "exemplo.com". Visualmente, o mapa é muito parecido com o da resolução padrão. A diferença aqui são os registros PTR, no mais, a sintaxe permanece a mesma. Após toda e qualquer alteração realizada nos arquivos de configuração existentes, é necessário reiniciar o serviço de rede. O leitor verá que este comportamento é o mesmo para quase a totalidade dos serviços de rede.

Isto feito, podemos consultar e iniciar a análise do serviço para constatar se ele foi bem configurado e se está de acordo com o que foi desejado. Existem duas ferramentas populares para realizar ações de diagnóstico em servidores DNS, sendo elas a *dig* (*Domain Information Groper*) e a *nslookup*. Neste curso usaremos a *nslookup*, por ser uma ferramenta que pode ser encontrada na grande maioria dos sistemas operacionais modernos.

A análise é feita na linha de comando e a sintaxe você pode ver logo a seguir. Observe que na primeira linha vem o comando, buscando resolver o nome em questão. Após, pode-se visualizar a informação do endereço do servidor DNS e abaixo, o endereço e porta (53). Na segunda parte da saída do comando, pode-se visualizar o nome que foi solicitado e o endereço IP que está relacionado a ele no servidor.

```
$ nslookup www.example.com
Server:         192.168.1.1
Address:        192.168.1.1#53

Non-authoritative answer:
Name:   www.example.com
Address: 192.168.1.3
```

A análise pode ser feita de modo reverso como já foi dito. Neste caso, o cliente quer saber que nome está relacionado com determinado IP. Nos comandos a seguir, pode-se visualizar um exemplo deste tipo de solicitação. A diferença para solicitação padrão é que, ao invés de solicitarmos informações de um nome, são solicitadas informações acerca de um IP, e o resultado é um nome.

```
$ nslookup 192.168.1.3
Server:         192.168.1.1
Address:        192.168.1.1#53

Non-authoritative answer:
1.1.168.192.in-addr.arpa        name:    www.example.com
```

Existem várias outras técnicas que fazem parte dos sistemas de resolução de nomes, como: a troca de mapas de domínios entre servidores DNS; a segurança envolvida no processo, ilustradas pelo uso de ACLs (*Access Control Lists*); o armazenamento de mapas em sistemas de diretórios; etc. Para mais informações, vi*site* o *site* oficial do Bind (BIND, 2011).

Acompanhe o Casos e relatos a seguir e confira como um ponto pode fazer falta na configuração do mapa de domínio.

CASOS E RELATOS

A importância do ponto

Luciano é administrador de sistemas de uma instituição sem fins lucrativos. Há alguns dias ele decidiu montar um servidor de nomes (DNS) e registrar um domínio para desenvolver um *site* que desse uma identidade à instituição. Registrou o domínio da Registro.BR e, na hora de vincular os dois endereços como servidores primário e secundário, optou por servir somente com o primário, utilizando um serviço gratuito da Internet. Assim, ele estruturou o Bind9 como servidor primário na zona master e configurou o mapa do domínio. Tudo estava ocorrendo normalmente, porém, alguns dias depois dele ter desenvolvido o *site,* que estava hospedando no servidor web, percebeu que toda vez que ele tentava acessar o endereço “www.instituicao.com.br”, aparecia na barra de status do seu *browser* o endereço “www.insituicao.com.br.instituicao.com.br”. Investigou o problema, mas não conseguia achar a solução. Pediu, então, ajuda a André, um amigo que tinha mais experiência em servidores de redes, enviando-lhe os arquivos de configuração para que analisasse.

Analisando os arquivos, André percebeu que na linha 18 do código faltava um ponto. A informação aparecia como “www IN CNAME instituicao.com.br”. André então ligou para Luciano e pediu que ele alterasse o domínio, colocando o ponto depois do “BR”, de maneira que o domínio ficasse “www IN NAME instituicao.com.br.” porque, como a árvore conceitual do DNS inicia no ponto, ela obrigatoriamente deve terminar no “.”, principalmente quando for citar nomes dentro dos mapas DNS. Luciano arrumou o domínio e conseguiu acessar corretamente.

RECAPITULANDO

Neste capítulo, você estudou um conteúdo teórico necessário para a fixação dos conceitos relativos ao serviço de rede denominado Sistema de Nomes de Domínios ou *Domain Name System* (DNS). Nesse conteúdo, você conheceu o histórico do serviço e as necessidades existentes na época em que o serviço foi concebido. Conheceu, também, o modo de operação hierárquico do serviço, tendo uma visão sistêmica do serviço, e conferiu uma tabela com os tipos de registros mais utilizados em servidores DNS. Viu, ainda, toda a parte de instalação, configuração e análise de um servidor DNS baseado no Bind9, da ISC. No próximo capítulo, você estudará a importância do servidor de Proxy. Até mais!

Servidor Proxy

4

Este capítulo tratará de assuntos relacionados a servidores proxy, passando pela história do serviço, os tipos de servidores proxy existentes e os usos mais comuns desta tecnologia. Os serviços de proxy são de extrema importância para empresas de qualquer porte, pois em todos os tipos de corporações, há a necessidade de restrições em determinados serviços. Para encerrar este capítulo, você acompanhará um exemplo de instalação e configuração do serviço, usando o *Squid-Cache* como base, sobre a plataforma *Debian Linux/GNU*.

Ao final deste capítulo, você terá subsídios para:

a) conhecer a evolução e a história dos servidores proxy;

b) entender os tipos de proxy existentes no mercado;

c) compreender as principais utilizações dos servidores proxy;

d) conhecer as implementações de serviços proxy.

4.1 HISTÓRIA

A importância de utilizarmos servidores proxy, atualmente, na Internet é alta, devido à grande quantidade de informações na rede, à origem do conteúdo disponibilizado e posterior acesso, à velocidade dos *links* de comunicação, entre outros. Basicamente, podemos definir um servidor proxy como uma máquina que atente requisições e as passa para frente. Este servidor tem a função de intermediar conexões entre clientes e servidores. Por exemplo: um usuário solicita ao servidor proxy um conteúdo determinado de uma página na Internet e o servidor Proxy vai até a página, "pega" o conteúdo e o fornece ao usuário. Observe a figura a seguir.

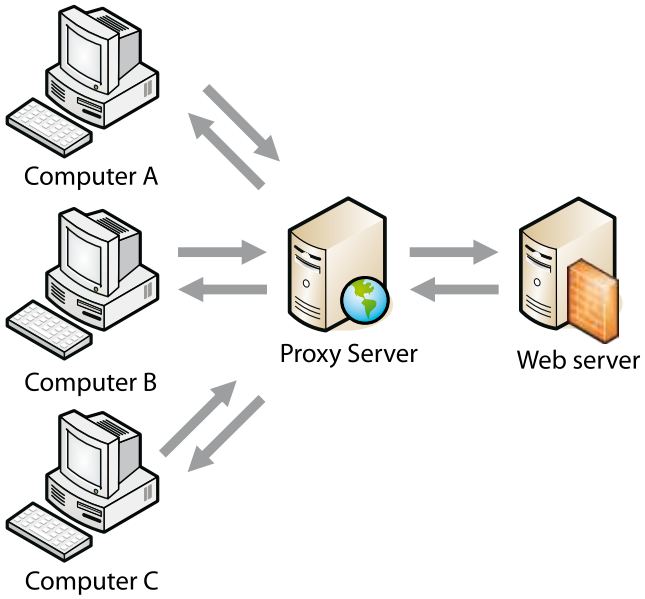


Figura 5 - Visão esquemática dos servidores proxy

Várias empresas, nos dias de hoje, estão procurando implantar soluções para controlar o tráfego gerado em suas redes por seus funcionários, buscando encontrar um tráfego de dados não correspondente às funções de seus empregados. Esta busca das empresas pelo controle de suas redes tem várias origens, entre elas podemos citar o aumento de produtividade da equipe, maior objetividade no ambiente de trabalho, e ainda, o monitoramento de seus funcionários.

Atualmente, utilizando-se de servidores Proxy com algumas ferramentas, é possível gerar gráficos de produtividade por funcionário, apontando a quantidade de horas que ele passou de fato no sistema da empresa trabalhando para os interesses da empresa e a quantidade de horas que ele passou, por exemplo, conversando com seus amigos no Facebook, Gtalk ou afins.

É importante notar que a prática de monitoramento de comportamento de funcionários na rede é um assunto controverso e muito discutido por advogados e juristas, porém, como é sabido, muitas empresas fazem uso destas tecnologias para suportar até decisões como promoções, ou mesmo, demissões. Como você verá no decorrer do capítulo, muitas empresas utilizam-se de proxy transparente, desta forma, os funcionários nem sabem que estão sendo monitorados.

O surgimento dos servidores proxy vêm da necessidade de interligar redes de computadores locais (*Local Area Network*) à Internet. No passado, e ainda nos dias de hoje, os *links* de comunicação eram lentos e a utilização de uma máquina que intermediasse o tráfego, fazendo *cache* dos dados já baixados era necessária para o aumento do desempenho da rede. Outro motivo citado é que os computadores clientes da rede interna das organizações não possuíam endereços válidos para a Internet e, devido a isto, eles precisariam ter um servidor que intermediasse as conexões.

Com o tempo, o desempenho dos *links* de comunicação foi aumentando e as demandas relativas a uma máquina que intermediasse o tráfego se tornaram vitais para controle do que é trafegado na rede das organizações. Além disso, as ferramentas que implementam servidores proxy foram melhorando e hoje temos ferramentas robustas que em conjunto com potentes analisadores de registros de conexões (logs) podem fornecer informações detalhadas sobre o comportamento de cada usuário na rede.

Agora que você já conhece a história dos servidores proxy, conheça os três tipos mais conhecidos, no próximo item. Continue atento e bom estudo!

4.2 TIPOS DE PROXY

Existem pelo menos três tipos conhecidos de proxy: *Forward Proxy* (Proxy de Encaminhamento), *Open Proxy* (Proxy Aberto) e *Reverse Proxy* (Proxy Reverso). Cada um deles tem uma função bem definida e uma arquitetura e modos de funcionamentos diferentes.

Os *Forward Proxy* são servidores onde os clientes utilizam um servidor para conectar-se à Internet. Eles são capazes de recuperar informações a partir de uma ampla variedade de fontes e, na maioria dos casos, de qualquer lugar da Internet. O termo Proxy de Encaminhamento é, na verdade, uma descrição do comportamento deste tipo de solução. Na figura a seguir, você pode ver a ilustração da arquitetura desse tipo de proxy.

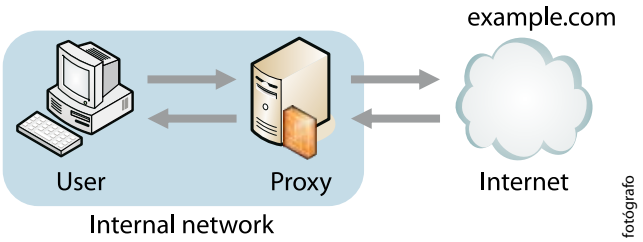


Figura 6 - Exemplo de Forward Proxy

Outro tipo de servidor é o Proxy Aberto. Este pode ser caracterizado por um servidor que é acessível a qualquer usuário da Internet. Comumente, este tipo de tecnologia permite aos usuários navegar anonimamente na Internet. A vantagem deste tipo de proxy é que o servidor "esconde" ou oculta o endereço IP do cliente que o está utilizando, dando assim um nível de anonimidade às solicitações para a Internet. Na grande maioria dos casos, eles são usados para acessar conteúdo não-autorizado.

Certamente, existem mais desvantagens do que vantagens na utilização deste tipo de tecnologia. A saber, governos mais repressivos utilizam e monitoram estes servidores durante 24 horas por dia, buscando identificar os usuários, para depois processá-los. Outros organismos, tanto aqui no Brasil como em outros países, os monitoram pelo fato de que muitos criminosos fazem uso deste tipo de abordagem para acessar, por exemplo, tráfico de drogas, armas e pessoas. Desta forma, é altamente aconselhado que não se utilize este tipo de tecnologia.

Veja, na figura a seguir, o exemplo da arquitetura deste tipo de abordagem.

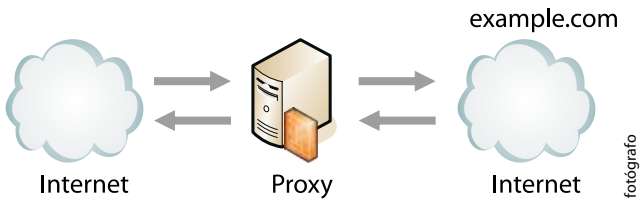


Figura 7 - Exemplo de Proxy Aberto

Por fim, há o tipo de proxy entitulado Proxy Reverso. É caracterizado como um servidor que aparece para o cliente como sendo um servidor comum, por exemplo, um servidor Web, porém, as solicitações são encaminhadas para um ou mais servidores de origem, que processam a solicitação. A resposta da solicitação é retornada ao cliente como sendo o servidor proxy que tivesse respondido. Na figura a seguir, é possível visualizar uma arquitetura de servidores de Proxy Reverso. Nela, o servidor proxy está intermediando as conexões que serão processadas pelo servidor Web.

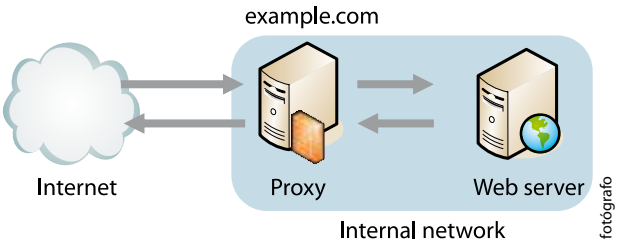


Figura 8 - Exemplo de Proxy Reverso

Há vários motivos e razões para a utilização de servidores de Proxy Reverso, como:

a) criptografia e/ou aceleração SSL (*Secure Socket Layer*) – quando *sites* seguros são criados, a criptografia SSL, muitas vezes, não é feita pelo servidor Web em si, mas por um proxy reverso, que é equipado com *hardware* de aceleração SSL;

b) balanceamento de carga – o proxy reverso pode distribuir a carga de vários servidores Web, sendo que cada um destes servidores age isoladamente com suas aplicações. Entretanto, devido à integração dada pelo proxy, os mesmos agem como se fossem um único servidor. Em alguns casos, o proxy reverso pode ter de reescrever as URLs em cada página Web;

c) servir conteúdo de cache estático – um proxy reverso pode evitar sobrecarga nos servidores Web usando mecanismos de *caching* (o ato de fazer *cache*) de conteúdo estático, como imagens e conteúdo estático em geral;

d) compressão – o servidor proxy pode otimizar e comprimir o conteúdo para acelerar o tempo de *download* do conteúdo;

e) segurança – o servidor proxy é uma camada adicional de defesa e pode proteger contra alguns ataques aos servidores Web, no entanto, não prevê qualquer proteção a ataques contra a aplicação Web ou o serviço em si, que é geralmente considerada a maior ameaça.

Agora que você já conhece os três tipos de servidores proxy, conheça onde eles são usados. Continue seus estudos!

4.3 USOS PARA SERVIDORES PROXY

Na maioria dos casos, os servidores proxy são utilizados para dois fins: *cache* e filtragem. Em outros casos, eles podem ser usados para registro (*logging*), monitoramento de redes, *gateway* para redes privadas, acessos anônimos, entre outros.

Neste momento, vamos nos concentrar na filtragem e *cache*. A filtragem realizada pelos servidores proxy é basicamente de conteúdo web. Ela proporciona importantes níveis de controles administrativos sobre o conteúdo que pode ser transmitido por meio do proxy. É geralmente usada em organizações que necessitam controlar o conteúdo acessado para verificar se ele está de acordo com a política de uso da rede.

Um servidor de proxy para filtragem de conteúdo, na maioria das vezes, suporta mecanismos de autenticação vinculados a ele. Isto é importante na identificação dos usuários para, posteriormente, cruzar as informações de acesso. Este tipo de servidor também produz uma extensa quantidade de *logs* (registros) de acessos a endereços específicos ou para monitorar a largura de banda utilizada em determinado momento.

Nesta etapa de filtragem é muito comum o administrador da rede incorporar um esquema de *blacklist* (lista-negra) e *whitelist* (lista-branca). A primeira é utilizada para determinar quais conteúdos não devem ser acessados pelo servidor proxy. Este bloqueio pode ser feito por: palavra-chave, URL, tipo de dados, tipo de protocolo, entre outras possibilidades. A segunda lista é utilizada para determinar qual URL não deve ser filtrada em nenhum momento. Geralmente um endereço contido em uma *whitelist* é um endereço reconhecidamente seguro.

Um servidor proxy de *cache* tem a função de acelerar as solicitações dos clientes, recuperando o conteúdo gravado a partir de um pedido que foi realizado anteriormente pelo mesmo cliente ou por outro cliente. Este tipo de servidor mantém cópias locais de conteúdos frequentemente solicitados, permitindo assim, que se reduza significativamente o uso dos *links* de comunicação. O servidor proxy para *cache* foi o primeiro tipo de servidor desta natureza.

Os servidores proxy de *cache*, além de terem sido os primeiros servidores proxy inventados, ainda são os mais populares, por sua facilidade de instalação e agregação na infraestrutura computacional de uma organização. Possuindo uma base de autenticação de usuários, é relativamente fácil incorporar um servidor de proxy transparente para controlar o acesso à Internet, emitir relatórios, entre outras funções.

O servidor Proxy ainda pode ser muito útil para ajudar no aumento de produtividade. Quer ver como? Confira o Casos e relatos a seguir.

CASOS E RELATOS

Proxy para aumento de produtividade

Em um escritório de advocacia, muitos funcionários, ao invés de usar os computadores para trabalho, estavam usando para acessar *sites* de relacionamento, como Orkut, Facebook e outros. Houve, então, a necessidade de controlar o acesso e bloquear alguns *site*s. Para isso, o gerente contratou uma empresa especializada em servidores de rede para implementar alguma solução que pudesse dar a eles uma visão geral do que os funcionários faziam, com possibilidade de relatórios para que pudessem identificar alguns *sites* como proibidos. Depois de algum tempo estudando a infraestrutura, que tinha apenas uma linha ASDL conectada a um modem, onde este enviava a conexão para um *switch* que fazia entrega dos IPs para os cliente, percebeu-se que o escritório não tinha nenhuma gerência sobre o ambiente. A solução foi a compra de um servidor se servisse como servidor DHCP, *firewall* e servidor Proxy. A empresa estruturou um *Squid Proxy Server* para filtrar todo o conteúdo. Foi realizado, ainda, um esquema de autenticação, baseado em LDAP para definir os usuários. Para o bloqueio dos *site*s indevidos, os consultores estruturaram um esquema de lista negra de domínios, e listaram os *site*s de relacionamento em questão. Como resultado desse processo todo, constatou-se que a produtividade aumentou 18% com a implantação do serviço, melhorando o nível de satisfação dos clientes do escritório, que agora tinham um tempo de espera um pouco menor.

Confira, agora, como os servidores proxy são implementados.

4.4 IMPLEMENTAÇÕES DE SERVIDORES PROXY

Dentre as implementações de servidores proxy existentes no mercado, elas se dividem em Proxy Web e Proxy Transparente. A primeira delas trata somente operações para a Web, como páginas, conteúdos e serviços. A segunda, pode tratar também de conteúdos Web, mas vai além, pois trabalha de modo que nenhum cliente tenha ciência de que está sendo monitorado ou que suas conexões estão sendo filtradas.

O Proxy Web, por definição, é o tipo de proxy que se concentra em filtrar o tráfego da *World Wide Web* (www). Geralmente, o seu uso é para servir como um proxy *cache* para páginas Web. Em sua grande maioria, os servidores proxy que têm esta função implantam operações de *blacklist* e *whitelist* em sua arquitetura. Este tipo de tecnologia é muito utilizada em ambientes educacionais, empresariais, bibliotecas e em qualquer outro tipo de ambiente que neces*site* de filtragem de conteúdo.

Existem ainda alguns tipos de servidores proxy que adequam às páginas Web fornecidas de modo a configurá-las para celulares, iPods, iPads e afins. Este tipo de servidor proxy está sendo muito utilizado atualmente devido ao grande aumento na utilização de aparelhos portáteis para acesso à Internet. Um exemplo deste tipo de arquitetura pode ser visualizada na figura a seguir.

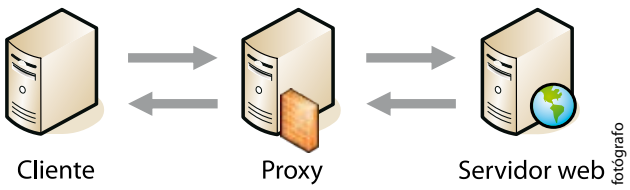


Figura 9 - Exemplo de arquitetura de Servidores Proxy Web

O segundo tipo de implementação de proxy é o transparente. Os servidores proxy transparente têm a função de interceptar a comunicação que vai para a Web sem a necessidade de alterações nas configurações das estações de trabalho. Os clientes não sabem ou não precisam saber que estão "atrás" de um servidor proxy. As definições que estão na RFC 2616 (*Hypertext Transfer Protocol* – HTTP/1.1) dizem que o proxy transparente é um tipo de proxy que não modifica o pedido ou a resposta das solicitações dos clientes.

O objetivo da utilização deste tipo de tecnologia é monitorar redes sem a necessidade de que os clientes destas saibam que estão sendo monitorados. É largamente utilizado em redes corporativas a fim de saber os hábitos de seus funcionários, para ver se estes se adequam com a política de uso da rede. Ainda, este tipo de proxy tem o objetivo de aliviar a carga da rede, já que pode trabalhar como servidor de *cache*, melhorando as taxas de *download* e *upload*.

Em sua maioria, este tipo de proxy trabalha de forma simples, com direcionamento de portas. Ele captura todo o tráfego direcionado para a porta 80 (HTTP) do *gateway* da rede e envia para a porta padrão do servidor proxy, que registra e efetua as operações que estiverem configuradas pra realizar. Depois disto, ele direciona a mensagem para o endereço de destino. Com essa simples regra de direcionamento, ele garante que todo o tráfego que for para a Internet será registrado pelo servidor proxy.

FIQUE ALERTA

Comumente, a tarefa de redirecionamento de portas é realizada no *firewall* da organização. Algumas implementações integradas, como o ISA Server da Microsoft, permitem uma configuração integrada, porém, em sua maioria, as ferramentas não têem autonomia sobre redirecionamento de tráfego na rede. Elas agem de modo passivo, aguardando que o *firewall* encaminhe as solicitações para ela.

Depois de conferir como são implementados os servidores proxy, que tal conhecer alguns exemplos? Então, continue atento e siga em frente com seus estudos.

4.5 EXEMPLOS DE SERVIDORES PROXY

No quadro a seguir, encontram-se alguns exemplos de aplicações que implementam o serviço de proxy. Elas vão desde servidores HTTP convencionais até servidores proxy reversos com funções de multiprotocolos e NAT (*Network Address Translate*). Observe que existem aplicações tanto para sistemas operacionais da família Windows como para Linux e UNIX em geral.

SOLUÇÃO	DESCRIÇÃO
Apache Traffic Server	Solução de alta performance *open-source* para implantação de servidores proxy HTTP baseados no servidor web Apache.
lighttpd	Servidor Web de código-fonte aberto. É otimizado para velocidade em ambientes críticos.
Nginx	Servidor leve, de alto desempenho para servidor web proxy, proxy reverso e *e-mail* proxy (IMAP/POP3).
Polipo	Trata-se de um servidor de proxy cache leve para servidor Web.
Privoxy	Servidor proxy que tem como alvo a segurança.
Tinyproxy	Servidor pequeno e rápido que suporta HTTP, proxy reverso e proxy transparente.
Verniz	Servidor de proxy reverso de código-aberto.
WinGate	Servidor proxy multiprotocolo que suporta proxy *cache*, *firewall* e NAT para plataforma Windows.
Microsoft Forefront Threat Management Gateway (ISA)	Servidor da Microsoft que implementa proxy de encaminhamento (*forward*) e proxy *cache*, e ainda funciona como proxy reverso e *firewall*.

Quadro 3 - Exemplos de servidores proxy

Nesse item, você conheceu a descrição de cada solução de aplicação dos servidores Proxy. No item seguinte, você acompanhará a instalação e a configuração de um servidor Proxy.

4.6 INSTALAÇÃO E CONFIGURAÇÃO DE UM SERVIDOR PROXY

Neste curso usaremos como base o servidor Proxy *Squid*, que suporta os protocolos HTTP, HTTPS, FTP, entre outros. Com sua utilização, ele reduz o consumo de largura de banda e melhora os tempos de resposta, utilizando um sistema de *cache* interno, baseando-se em páginas que são frequentemente acessadas via Web. Há, ainda, a possibilidade de utilizá-lo como proxy reverso (SQUID, 2011).

O *Squid* foi desenvolvido originalmente para ser executado em sistemas operacionais do tipo Unix-Like, tais como AIX, HP-UX, Linux, entre outros. Com o tempo, ele foi melhorado e hoje suporta uma ampla gama de plataformas. As plataformas computacionais suportadas pelo *Squid* contêm:

a) AIX;

b) BSDI;

c) Digital Unix;

d) FreeBSD;

e) HP-UX;

f) IRIX;

g) Linux;

h) Mac OS X;

i) NetBSD;

j) NeXTStep;

k) OpenBSD;

l) OS/2 and eComStation;

m) SCO OpenServer;

n) Solaris;

o) UnixWare;

p) Windows.

SAIBA MAIS

É possível usar o *Squid Proxy Server* para plataforma Windows. Veja mais informações em: <www.squid.org>.

Acompanhe como é feita a instalação.

4.6.1 INSTALAÇÃO

O sistema operacional que utilizaremos como base para os testes de instalação e configuração será o Debian Linux/GNU, versão Squeeze. A instalação pode ser feita utilizando o gerenciador de pacotes do Debian, o "apt-get", ou acessando o *site* oficial do *Squid*, baixando o código-fonte e instalando o mesmo. Confira, a seguir, um exemplo de instalação via gerenciador. (SQUID, 2011).

```
root@server:/home/douglas# apt-get install squid
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
Os pacotes extra a seguir serão instalados:
  squid-common squid-langpack
Pacotes sugeridos:
  squidclient squid-cgi logcheck-database resolvconf
smbclient winbind
Os NOVOS pacotes a seguir serão instalados:
  squid squid-common squid-langpack
0 pacotes atualizados, 3 pacotes novos instalados, 0 a
serem removidos e 2 não atualizados.
É preciso baixar 1398 kB de arquivos.
Depois desta operação, 8315 kB adicionais de espaço em
disco serão usados.
Você quer continuar [S/n]? S
Obter:1 http://debian.pop-sc.rnp.br/debian/ squeeze/
main squid-langpack all 20100628-1 [228 kB]
Obter:2 http://debian.pop-sc.rnp.br/debian/ squeeze/
main squid-common all 2.7.STABLE9-2.1 [352 kB]
Obter:3 http://debian.pop-sc.rnp.br/debian/ squeeze/
main squid amd64 2.7.STABLE9-2.1 [818 kB]
Baixados 1398 kB em 0s (9167 kB/s)
Pré-configurando pacotes ...
Selecionando pacote previamente não selecionado squid-
-langpack.
(Lendo banco de dados ... 23065 ficheiros e directórios
actualmente instalados.)
```

```
  Desempacotando squid-langpack (de .../squid-langpa-
ck_20100628-1_all.deb) ...
  Selecionando pacote previamente não selecionado squid-
-common.
  Desempacotando squid-common (de .../squid-
common_2.7.STABLE9-2.1_all.deb) ...
  Selecionando pacote previamente não selecionado squid.
  Desempacotando squid (de .../squid_2.7.STABLE9-2.1_
amd64.deb) ...
  Processando gatilhos para man-db ...
  Configurando squid-langpack (20100628-1) ...
  Configurando squid-common (2.7.STABLE9-2.1) ...
  Configurando squid (2.7.STABLE9-2.1) ...
  Creating squid spool directory structure
  2011/09/14 12:42:12| Creating Swap Directories
  Restarting Squid HT Squid: squid.
```

Dado o comando que você acabou de conferir, toda a configuração do servidor proxy Squid em sistemas operacionais Debian Linux/GNU fica no diretório "/etc/squid". A seguir, você pode ver o conteúdo do diretório. Neste momento, o único arquivo existente é o "squid.conf". Este é o arquivo principal de configuração do servidor proxy. Neste diretório, podemos ainda colocar os arquivos de restrições de *blacklist,* ou ainda, as exceções das *whitelists*.

```
  root@server:~# ls -l /etc/squid/
  total 172
  -rw------- 1 root root 169404 Set 14 12:42 squid.conf
```

O conteúdo do arquivo "squid.conf" no momento da instalação é de mais de 4000 linhas. É um arquivo muito bem documentado, prevendo a maioria das possibilidades de uso da aplicação. Não é nosso objetivo aqui explicar linha a linha desta configuração. O leitor pode ler o arquivo de configuração por si só, e definir como deseja que seu servidor proxy se comporte, porém, no comando a seguir, é dado um arquivo "squid.conf" inicial, que pode ser usado para dar início à configuração do servidor.

```
  # O parâmetro http_port é o número da porta a ser uti-
lizada pelo Squid.
  http_port 3128
  # O parâmetro visible_hostname é o nome de nosso ser-
vidor.
  visible_hostname server
  # As ACL (Access Control List) são as regras que definirão
quem poderá # usar nosso proxy, no caso abaixo são todos.
  acl all src 0.0.0.0/0.0.0.0
  http_access allow all
```

VOCÊ SABIA?

O servidor proxy Squid é muito bem documentado em seu *site* oficial (SQUID, 2011) e lá, podem ser encontradas todas as informações necessárias para a customização do servidor.

Com esta configuração mínima, nós já podemos iniciar as configurações de customização do Squid. Como você já sabe, não esgotaremos todas as possibilidades de configuração neste capítulo, pois são várias, mas fica a dica para você pesquisar e se aprofundar mais no assunto.

RECAPITULANDO

Nesse capítulo, você conheceu o servidor Proxy. Conferiu a parte teórica que envolve o serviço, como a história do mesmo e seus motivos e necessidades que levaram a criação da tecnologia. Na sequência do capítulo, você conferiu os tipos de servidores proxy e as suas possíveis utilizações, e acompanhou a demonstração de uma instalação e configuração mínima utilizando o servidor Proxy Squid. Ainda tem muita informação pela frente!

Servidor Web

5

O tema deste capítulo é o servidor Web. Este é um tipo popular de servidor que tem a função primária de entregar páginas web para clientes que solicitaram determinado conteúdo. Em termos práticos, isto significa que quando um determinado cliente fizer uma solicitação para o servidor web, ele fornecerá páginas (HTML, PHP, ASP, JSP e afins) e conteúdos adicionais, tais como: imagens, documentos, scripts e afins. Sua compreensão acerca dos aspectos técnicos e práticos relacionados é muito importante para que o administrador de redes possa implementar o serviço da melhor maneira possível.

Ao final deste capítulo, você terá subsídios para:

a) conhecer a evolução e a história dos servidores Web;

b) entender as características comuns dos servidores Web;

c) compreender URLs;

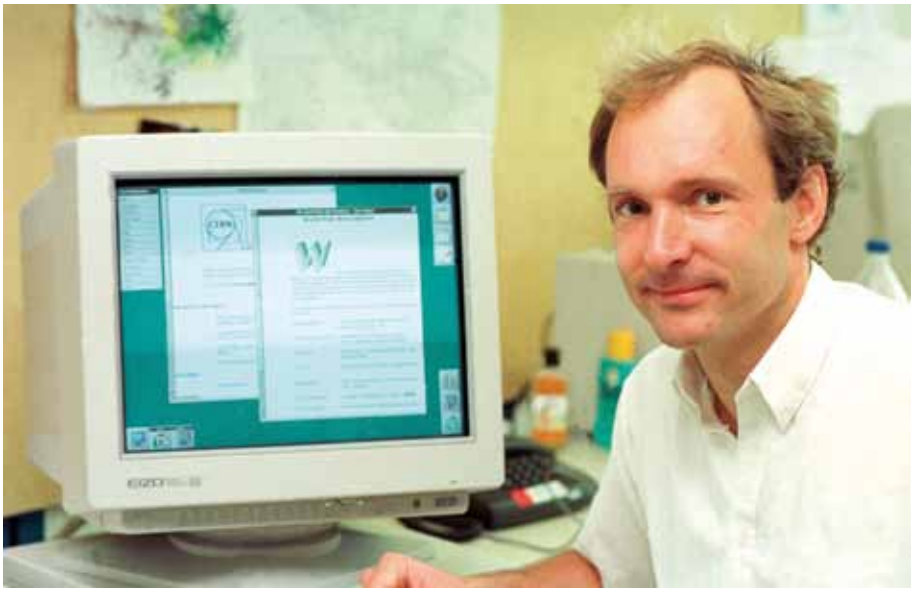
d) conhecer a segurança envolvida para este tipo de servidor;

e) entender como é estruturado o mercado de servidores Web.

Preparado para começar? Então, vamos lá!

5.1 HISTÓRIA

Tim Berners-Lee, um cientista do CERN (*European Organization for Nuclear Research*), inventou a *World Wide Web* (WWW) em 1989. A Web foi originalmente concebida e desenvolvida para atender a demanda de compartilhamento de informações automática entre os cientistas que trabalhavam em diferentes universidades e institutos de todo o mundo. "A ideia básica da WWW era fundir as tecnologias de computadores pessoais, redes de computadores e hipertextos em um sistema de informação poderoso e fácil de usar globalmente". (CERN, 2011).



fotógrafo

Figura 10 - Tim Berners-Lee, o inventor da WWW

O projeto de Berners-Lee resultou em dois programas, em 1990:

a) um navegador chamado *World Wide Web*;

b) um servidor da Web do mundo em primeiro lugar, mais tarde conhecida como CERN httpd, o qual decorreu posteriormente como o servidor Web chamado NeXTSTEP .

Entre 1991 e 1994, a simplicidade e a eficácia das tecnologias de início utilizadas para navegar e trocar dados por meio da *World Wide Web* ajudaram a portá-los para muitos sistemas operacionais diferentes e difundir o seu uso entre os grupos socialmente diversificados de pessoas: primeiro em organizações científicas, depois em universidades e, finalmente, na indústria. Em 1994, Tim Berners-Lee decidiu constituir a *World Wide Web Consortium* (W3C) para regular o desenvolvimento das diversas tecnologias envolvidas (HTTP, HTML, etc.) por meio de um processo de normalização. (WEB SERVER, 2011).

Quando nós falamos “clientes”, geralmente estamos falando de navegadores web, tais como Mozilla, Firefox, Chrome, Safari, etc. Na prática, um usuário abre um navegador web e solicita determinado endereço de uma página. A comunicação se inicia e a primeira etapa é consultar o servidor DNS a fim de determinar o endereço IP do destino. Conseguindo esta informação, por padrão, será disparada uma requisição na porta 80 do servidor web, que responderá com o conteúdo solicitado.

A figura a seguir permite a visualização de uma arquitetura padrão de servidores Web.

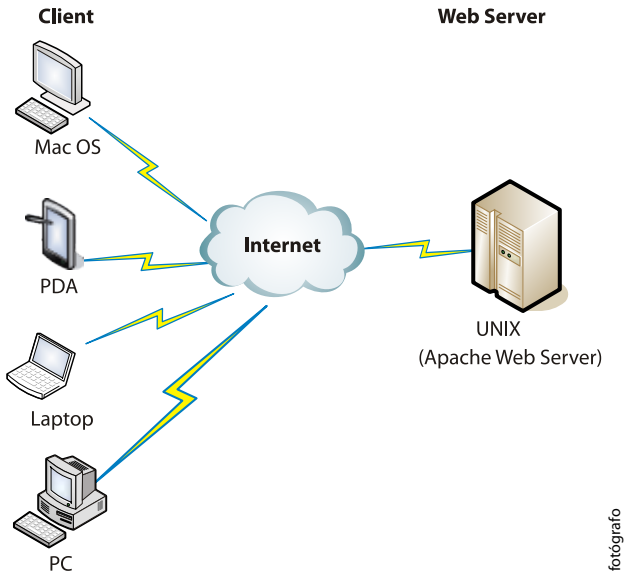


Figura 11 - Exemplo de arquitetura de servidores Web

Hoje em dia, há um tipo de cliente chamado *web crawler*. Por definição, estes mecanismos são *softwares* autônomos com a função de “vasculhar” a Internet, mapeando recursos disponíveis. Eles são muito utilizados por mecanismos de buscas, tais como Google, Altavista, Yahoo, entre outros. Desta forma, estes mecanismos mapeiam a Internet e quando um cliente solicita uma busca por determinado termo, eles saberão onde está a informação desejada.

Como função principal, o servidor web serve conteúdo, porém, a implementação do protocolo HTTP também inclui formas de receber conteúdo de clientes. Este recurso pode ser utilizado em formulários eletrônicos nas páginas, *upload* de arquivos para o servidor, entre outras características. A maioria dos servidores web modernos suportam linguagens de programação baseadas em “scritps” para proporcionar dinamicidade às páginas web. Dentre estas linguagens, podemos citar: ASP (*Active Server Pages*), PHP (*Hypertext Preprocessor*), CGI (*Common Gateway Interface*), JSP (*Java Server Pages*), entre outras linguagens atuais.

Estas linguagens de programação, além de darem dinamicidade às aplicações que serão disponibilizadas pelo servidor, proporcionam, ainda, uma camada de comunicação com servidores de bancos de dados, desta forma, provendo possibilidades inimagináveis desde a época em que a tecnlogia foi criada. Hoje temos complexos sistemas web que gerenciam operações financeiras, como em bancos, financeiras, bolsas de valores, entre outros. Esta evolução dos servidores web levou a Internet a um nível nunca imaginado por seus idealizadores.

Mas, os servidores web nem sempre são usados somente para servir a WWW, pois eles também podem ser encontrados em dispositivos como impressoras, roteadores, *switchs*, *webcams*, etc., fornecendo uma interface de comunicação com estes dispositivos. Neste caso, o servidor web pode servir tanto para monitorar os equipamentos como para administrar o dispositivo, fornecendo, desta forma, uma interface de configuração mais "amigável" para o cliente.

5.2 CARACTERÍSTICAS COMUNS

Em se tratando de servidores Web, algumas características são comuns entre eles. São características básicas e não determinam a totalidade do que este tipo de tecnologia pode proporcinar mas, naturalmente, norteiam as definições, como a hospedagem virtual (*virtual hosting*), suporte a arquivos grandes, largura de banca limitada e suporte a linguagens de *script*. (WEB SERVER, 2011).

A hospedagem virtual pode ser definida como a possibilidade do servidor Web de hospedar muitos *sites* usando um único endereço IP, ou ainda, a possibilidade do servidor hospedar o mesmo *site* usando múltiplos endereços IPs. Esta é uma característica básica que está presente na maioria dos servidores Web populares.

Sobre suportar arquivos grandes, com o avanço da tecnologia isto se tornou um aspecto básico que todos os servidores devem suportar. A caraterística de limitar a largura de banca significa que o servidor web deve ser customizável o suficiente para poder controlar a quantidade de requisições que pode receber, a fim de não saturar a rede e o sistema computacional que está instalado.

A característica de suporte à *server-side scripting* para geração de páginas Web dinâmicas, atualmente, é considerada essencial para a operação do servidor Web. Hoje em dia, a maioria esmagadora dos *site*s disponibilizados na Web tem algum grau de dinamicidade, o que exige que o servidor suporte linguagens de programação Web, como PHP, ASP, CGI, JSP, entre outras.

SAIBA MAIS

Para maiores informações sobre a parte de *scripting* para a geração de páginas dinâmicas, acesse o *site* oficial do Apache Web Server em: <http://www.apache.org>.

5.3 URL

A URL (*Uniform Resource Locator* ou Localizador de Recurso Uniforme) tem importância fundamental na estrutura dos servidores Web. Este componente, basicamente, é um ponteiro para um objeto ou serviço na Internet (ou Intranet). Ele descreve como acessar um objeto por cinco componentes básicos:

a) protocolos: HTTP, HTTPS, FTP, LDAP, etc.;

b) nome do Host;

c) porta TCP/IP: 80, 443, 21, etc. (opcional);

d) diretório (opcional);

e) nome do arquivo (opcional).

Por exemplo:

> "http://www.exemplo.com/pasta/arquivo.html".

No exemplo acima, podemos determinar que o protocolo é o "http" e o nome do host é o "www.exemplo.com". A porta também está determinada, mas é opcional, visto que, por padrão, quando um cliente acessa um endereço "http" via *browser,* automaticamente o *browser* entende que a solicitação será realizada na porta 80 do servidor de destino. Após isto, vem o diretório, "pasta" e em seguida, o arquivo a ser acessado, chamado "arquivo.html". Desta forma, é identificada uma URL. No quadro a seguir, é possível ver alguns exemplos de protocolos suportados em URLs. Confira!

PROTOCOLO	DESCRIÇÃO
http	Acessa um arquivo remoto via protocolo HTTP.
https	Acessa um arquivo remoto via protocolo HTTP/SSL.
ftp	Acessa um arquivo remoto via protocolo FTP (*File Transfer Protocol*).
mailto	Envia *e-mail* para um endereço designado.
news	Acessa grupo de notícias Uset (está caindo em desuso).
telnet	Efetua *login* em um computador remoto.
Ldap	Acessa um serviço de diretórios LDAP.
File	Acessa um arquivo local.

Quadro 4 - Exemplos de protocolos que compõem URLs

Mas será que esse tipo de procotolo é seguro? Confira isso no item a seguir.

5.4 SEGURANÇA

O protocolo HTTP, em sua essência, não prevê uma camada de segurança de dados para garantir a confiabilidade das informações, desta forma, o projeto OpenSSL (OpenSSL, 2011) é um esforço colaborativo para desenvolver uma biblioteca de criptografia de propósito geral e a implementação dos protocolos *Secure Socket Layer* (SSL v2/v3) e *Transport Layer Security* (TLS v1).

O projeto é gerenciado por uma comunidade internacional de voluntários que usa a Internet para se comunicar, planejar e desenvolver o OpenSSL e a sua documentação relacionada. O OpenSSL é baseado na biblioteca SSLeavy desenvolvida por Eric A. Young e Tim J. Hudson. Esta tecnlogia é livre e você pode usar para propósitos comerciais ou não.

Como trata-se de uma biblioteca de criptografia de propósito geral, o OpenSSL pode ser "plugado" (integrado) a vários serviços da Internet. Entre os serviços que podem ser integrados com o OpenSSL estão: Apache Web Server, Postfix Mail System, Sendmail Mail System, Squid Proxy Server, entre outros. No caso dos sistemas de *e-mail*, é mais comum que esta integração use o TLS para transporte seguro das mensagens eletrônicas.

Quando um servidor Web é integrado ao OpenSSL, este abre um *socket* na porta 443 do servidor e esta será responsável por responder às solicitações dos clientes. Na etapa de instalação do servidor Web, nos próximos itens, veremos na prática como é possível implementar o protocolo HTTPS (HTTP sobre SSL) usando o OpenSSL somado ao servidor Apache Web Server.

5.5 ESTRUTURA DO MERCADO

Em março de 2011, a Netcraft, reconhecida empresa da Internet que faz pesquisas de mercado desde o início da Internet, realizou uma pesquisa de estrutura de mercado (*Market Share*) para avaliar quais são os servidores Web mais utilizados na Internet. (NETCRAFT, 2011). O resultado desta pesquisa pode ser encontrado na tabela a seguir.

Tabela 1 - Fatia de mercado dos Servidores Web

PRODUTO	DISTRIBUIDOR	*SITE*S	PORCENTAGEM
Apache	Apache	179,720,332	60.31%
IIS	Microsoft	57,644,692	19.34%
nginx	Igor Sysoev	22,806,060	7.65%
GWS	Google	15,161,530	5.09%
lighttpd	lighttpd	1,796,471	0.60%

Fonte: Netcraft (2011)

Acompanhe um exemplo sobre armazenamento no Casos e relatos. Boa leitura!

CASOS E RELATOS

Colaboração para armazenamento de criatividade

Uma empresa na área de marketing estava sofrendo há tempos com a perda de ideias criativas das equipes porque, em sua maioria, estas informações ficavam em *e-mails* ou documentes de texto espalhados pelos computadores pessoais. Desta forma, não havia nenhum nível de colaboração para trocar ideias *on-line*, tudo era feito pessoalmente, em reuniões, e o que não era anotado se perdia, o que se tornava um problema para a empresa. Então, o gerente de produção chamou o administrador da rede e expôs o problema e os desafios que estavam relacionados a ele. O administrador usou o exemplo da *Wikipédia*, que é uma plataforma de colaboração que foi desenvolvida para resolver um problema semelhante ao da empresa. Na *Wikipédia*, uma pessoa começa um determinado assunto e os outros podem contribuir para melhorar, e tudo fica registrado.

O gerente gostou da ideia pois era exatamente isso que estavam procurando. Dessa forma, o administrador de rede iniciou o processo de configuração de um servidor web e um banco de dados, os quais iriam suportar a aplicação da *Wikipédia*, que é chamada de *mediawiki*. Com tudo pronto, a ferramenta foi apresentada para os gerentes e funcionários que adoraram a ideia de poderem colaborar uns com os outros, almejando assim um crescimento conjunto para atingirem as metas.

5.6 INSTALAÇÃO DE UM SERVIDOR WEB

Para demonstrar a instalação de um servidor web, usaremos o Apache Web Server como padrão para a instalação. O Apache tem suporte a uma vasta gama de plataformas, entre elas: AIX, HP-UX, Linux, Windows. Utilizando esse servidor web, o conhecimento pode ser aplicado em outras plataformas. Como padrão, usaremos uma distribuição DEbian Linux/GNU Squeeze.

VOCÊ SABIA?

Você sabia que o nome Apache vem do trocadilho em inglês "A *pathy Server*" ou Um Servidor Remendado? Isto porque a equipe que desenvolveu o Apache era mantenedora do HTTP NCSA e eles forneciam vários *patch* (arquivos de correção) para o servidor Web da NCSA.

A instalação, assim como em outros serviços, pode ser feita utilizando o gerenciador de pacotes "apt-get", ou ainda, acessando o *site* oficial da ferramenta e fazendo o *download* do pacote. Nos comandos a seguir, você vê um exemplo de configuração do servidor Web Apache.

```
root@server:/# apt-get install apache2
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
Os pacotes extra a seguir serão instalados:
     apache2-mpm-worker  apache2-utils  apache2.2-bin
apache2.2-common  libapr1  libaprutil1  libaprutil1-dbd-
sqlite3 libaprutil1-ldap ssl-cert
Pacotes sugeridos:
    apache2-doc  apache2-suexec  apache2-suexec-custom
openssl-blacklist
Os NOVOS pacotes a seguir serão instalados:
   apache2 apache2-mpm-worker apache2-utils apache2.2-
bin apache2.2-common libapr1 libaprutil1 libaprutil1-dbd-
sqlite3 libaprutil1-ldap
   ssl-cert
0 pacotes atualizados, 10 pacotes novos instalados, 0 a
serem removidos e 2 não atualizados.
É preciso baixar 255 kB/2168 kB de arquivos.
Depois desta operação, 7381 kB adicionais de espaço em
disco serão usados.
Você quer continuar [S/n]? S
Obter:1  http://debian.pop-sc.rnp.br/debian/  squeeze/
main libapr1 amd64 1.4.2-6+squeeze3 [94,2 kB]
Obter:2  http://debian.pop-sc.rnp.br/debian/  squeeze/
main libaprutil1 amd64 1.3.9+dfsg-5 [92,3 kB]
Obter:3  http://debian.pop-sc.rnp.br/debian/  squeeze/
main libaprutil1-dbd-sqlite3 amd64 1.3.9+dfsg-5 [28,1 kB]
Obter:4 http://debian.pop-sc.rnp.br/debian/ squeeze/
```

```
  main libaprutil1-ldap amd64 1.3.9+dfsg-5 [25,7 kB]
  Obter:5 http://debian.pop-sc.rnp.br/debian/ squeeze/
main ssl-cert all 1.0.28 [14,8 kB]
  Baixados 255 kB em 0s (546 kB/s)
  Pré-configurando pacotes ...
  Selecionando pacote previamente não selecionado liba-
pr1.
  (Lendo banco de dados ... 24705 ficheiros e directórios
actualmente instalados.)
  Desempacotando libapr1 (de .../libapr1_1.4.2-
-6+squeeze3_amd64.deb) ...
  Selecionando pacote previamente não selecionado liba-
prutil1.
  Desempacotando libaprutil1 (de .../
libaprutil1_1.3.9+dfsg-5_amd64.deb) ...
  Selecionando pacote previamente não selecionado liba-
prutil1-dbd-sqlite3.
  Desempacotando libaprutil1-dbd-sqlite3 (de .../liba-
prutil1-dbd-sqlite3_1.3.9+dfsg-5_amd64.deb) ...
  Selecionando pacote previamente não selecionado liba-
prutil1-ldap.
  Desempacotando libaprutil1-ldap (de .../libaprutil1-
-ldap_1.3.9+dfsg-5_amd64.deb) ...
  Selecionando pacote previamente não selecionado
apache2.2-bin.
  Desempacotando apache2.2-bin (de .../apache2.2-
-bin_2.2.16-6+squeeze3_amd64.deb) ...
  Selecionando pacote previamente não selecionado apa-
che2-utils.
  Desempacotando apache2-utils (de .../apache2-
-utils_2.2.16-6+squeeze3_amd64.deb) ...
  Selecionando pacote previamente não selecionado
apache2.2-common.
  Desempacotando apache2.2-common (de .../apache2.2-
-common_2.2.16-6+squeeze3_amd64.deb) ...
  Selecionando pacote previamente não selecionado apa-
che2-mpm-worker.
  Desempacotando apache2-mpm-worker (de .../apache2-mpm-
worker_2.2.16-6+squeeze3_amd64.deb) ...
```

```
  Selecionando pacote previamente não selecionado apa-
che2.
  Desempacotando   apache2   (de   .../apache2_2.2.16-
-6+squeeze3_amd64.deb) ...
  Selecionando pacote previamente não selecionado ssl-
-cert.
  Desempacotando ssl-cert (de .../ssl-cert_1.0.28_all.
deb) ...
  Processando gatilhos para man-db ...
  Configurando libapr1 (1.4.2-6+squeeze3) ...
  Configurando libaprutil1 (1.3.9+dfsg-5) ...
  Configurando libaprutil1-dbd-sqlite3 (1.3.9+dfsg-5) ...
  Configurando libaprutil1-ldap (1.3.9+dfsg-5) ...
  Configurando apache2.2-bin (2.2.16-6+squeeze3) ...
  Configurando apache2-utils (2.2.16-6+squeeze3) ...
  Configurando apache2.2-common (2.2.16-6+squeeze3) ...
  Enabling site default.
  Enabling module alias.
  Enabling module autoindex.
  Enabling module dir.
  Enabling module env.
  Enabling module mime.
  Enabling module negotiation.
  Enabling module setenvif.
  Enabling module status.
  Enabling module auth_basic.
  Enabling module deflate.
  Enabling module authz_default.
  Enabling module authz_user.
  Enabling module authz_groupfile.
  Enabling module authn_file.
  Enabling module authz_host.
  Enabling module reqtimeout.
  Configurando apache2-mpm-worker (2.2.16-6+squeeze3) ...
  Configurando apache2 (2.2.16-6+squeeze3) ...
  Configurando ssl-cert (1.0.28) ...
```

Após essa etapa, o servidor web Apache Web Server versão 2 estará instalado no sistema operacional. Nessa instalação padrão, os arquivos de configuração estarão armazenados no diretório "/etc/apache2". Uma série de diretórios e arquivos estão disponíveis para a customização do servidor. Confira, a seguir, uma descrição de cada um dos arquivos e pastas alocados no sistema.

a) **apache2.conf** – arquivo de configuração principal do servidor.

b) **conf.d** – pasta que contém configurações diversas do servidor.

c) **httpd.conf** – arquivo legado da versão 1.X do Apache. Atualmente é mantido por questões de compatibilidade.

d) **mods-available** – pasta que contém os módulos que podem ser "plugados" ao servidor Web para estender as características.

e) **mods-enabled** – pasta onde são *linkados* os módulos que estão na pasta "mods-available".

f) **ports.conf** – arquivo que contém as definições de portas do servidor Web. Geralmente são duas possíveis: porta 80 (HTTP), por padrão, e 443 (HTTPS), quando o módulo SSL estiver habilitado.

g) ***sites*-available** – pasta que contém os arquivos de configuração dos *sites* que podem estar presentes nas configurações do servidor Web.

h) ***sites*-enabled** – pasta que contém os *links* para a pasta *sites-available*, que habilitarão os *sites* criados.

FIQUE ALERTA

Não é porque um módulo ou arquivo de *site* está nas pastas *-available que eles estarão presentes nas configurações do servidor Web. Um *site* ou módulo precisa ser habilitado, sempre, inserindo-os nos diretórios *mods-enable* e/ou *sites-enable*.

A configuração do servidor Apache Web Server é bem simples e intuitiva. Em seu arquivo de configuração principal há uma vasta documentação sobre cada um dos parâmetros que podem ser habilitados e utilizados para customização do servidor. Não é objetivo, neste momento, descrever as mais de 200 linhas do arquivo de configuração, porém, você pode acessar o *site* oficial da ferramenta Apache e ter acesso às diversas configurações possíveis.

Entretanto, é importante salientar a parte de configurações de *sites* que pode ser feita criando um arquivo no diretório "/etc/apache2/*sites*-available" com o conteúdo apropriado. Após a criação do arquivo de configuração do *site* no diretório especificado, é fundamental *linkar* o arquivo usando a função de *links* simbólicos do Linux (ln) para a pasta *sites-enable*, habilitando, desta forma, o funcionamento do *site*. Feito isto, como a maioria dos serviços de rede, deve-se reiniciar o servidor. Acompanhe, a seguir, dois exemplos de configurações para *sites* HTTP e HTTPS.

*Site* **Exemplo.com (HTTP)**

```
  NameVirtualHost *:80

  <VirtualHost *:80>
      ServerAdmin douglas@exemplo.com
      DocumentRoot /srv/www/htdocs/exemplo/
      ServerName www.exemplo.com
      ErrorLog /var/log/apache2/exemplo-error_log
      CustomLog /var/log/apache2/exemplo-access_log com-
mon

      ServerAlias exemplo.com

      <Directory /srv/www/htdocs/exemplo>
           Options Indexes FollowSymLinks MultiViews
           AllowOverride All
           Order allow,deny
           allow from all
      </Directory>

  </VirtualHost>
```

 # *Site* Exemplo.com (HTTPS)

```
   NameVirtualHost *:443

   <VirtualHost *:443>
       ServerAdmin douglas@exemplo.com
       DocumentRoot /srv/www/htdocs/exemplo/
       ServerName www.exemplo.com
       ErrorLog /var/log/apache2/exemplo-https-error_log
      CustomLog /var/log/apache2/exemplo-https-access_log
common

       ServerAlias exemplo.com

       <Directory /srv/www/htdocs/exemplo>
            Options Indexes FollowSymLinks MultiViews
            AllowOverride All
            Order allow,deny
            allow from all
       </Directory>

   </VirtualHost>
```

Nesse comando é possível ver a configuração do *site* "www.exemplo.com", tanto para o protocolo HTTP como para o protocolo HTTPS. Nesse caso, o servidor Web está usando esquema de múltiplos *hosts* virtuais (*VirtualHost*) para realizar essa função. Isto quer dizer que o servidor Web responderá por requisições para ambos endereços: **"http://www.exemplo.com"** e **"https://www.exemplo.com"**.

a) Sobre as configurações possíveis de serem implementas em *hosts* virtuais, veja a seguir, uma descrição técnica para as diretrizes que foram dadas ao servidor no arquivo apresentado a pouco. O objetivo é nivelar os conhecimentos acerca das diretivas, norteando o aprendizado para soluções de problemas com servidores Web. NameVirtualHost: a diretiva **NameVirtualHost** é mandatória para casos de servidores Web que trabalham com múltiplos domínios baseados em nomes. Ela suporta IPs, nomes e portas, por exemplo: NameVirtualHost *:80. Desta forma, ela estará escutando em todas as interfaces de rede da máquina, na porta 80. É possível determinar um endereço IP específico para a diretiva, como por exemplo: NameVirtualHost 192.168.1.3:80.

b) **VirtualHost**: esta diretiva é utilizada agrupando várias outras diretivas. Sempre inicia-se com <VirtualHost> e termina-se em </VirtualHost>. Ela suporta nomes, IPs e portas em sua configuração inicial, como por exemplo: <VirtualHost *:80>. Desta forma, ela estará usando o NameVirtualHost :80 que foi criado anteriormente, porém, ela também pode receber parâmetros diretos de endereços IPs, como por exemplo: <VirtualHost 192.168.2.3:80>. Entre o grupo de diretivas que podem ser inseridas dentro de um *virtual host*, podemos citar as seguintes:

a) **ServerAdmin**: endereço de *e-mail* do administrador do *site*. Suporta endereços de *e-mail*, como por exemplo: douglas@exemplo.com;

b) **DocumentRoot**: esta diretiva determina o diretório que o servidor Web mostrará quando for acessar o *site*. Por exemplo: "DocumentRoot /srv/www/htdocs/exemplo", neste caso todas as vezes que algum usuário acessar o *site* "www.exemplo.com", o servidor Web exibirá o conteúdo do diretório especificado;

c) **ServerName**: esta diretiva determina o *hostname* do servidor Web. Ela é usada para criar o direcionamento à URL que deseja-se oferecer. Por exemplo: "ServerName www.exemplo.com", determina que este *site* virtual será acessado pelo nome "www.exemplo.com";

d) **ErrorLog**: esta diretiva determina em qual o arquivo o servidor Web irá armazenar os registros (*logs*) de erros que venham a acontecer com o conteúdo ou com o servidor relacionado;

e) **CustomLog**: esta diretiva é usada para apontar o arquivo onde o servidor Web irá armazenar os registros (*logs*) de acesso ao servidor. É muito utilizada para fazer análises de acessos ao servidor e gerar estatísticas;

f) **ServerAlias**: determina um nome alternativo para o servidor, como por exemplo: "ServerAlias exemplo.com". Quer dizer que toda vez que um usuário acessar, via *browser*, o endereço "exemplo.com" será a mesma coisa que digitar "www.exemplo.com".

c) **Directory**: esta diretiva é usada agrupando outras diretivas dentro dela. Sempre inicia-se a configuração com <Directory> e termina-se com </Directory> e tudo que estiver entre as diretivas fará parte da configuração de um determinado diretório. Essa diretiva é usada para configurar as opções de visualização, execução e controle de acesso ao diretório em questão.

a) **Options**: determina diretivas de controle sobre um diretório em particular. Ela pode ser definida como "None" para uma não configuração do diretório, porém, o mais comum é usarmos as seguintes funções:

b) **All**: todas as opções, exceto Multiviews são habilitadas;

c) **ExecCGI**: permite execução de scripts CGI dentro do diretório;

d) **FollowSymLinks**: o servidor irá "seguir" *links* simbólicos dentro do diretório;

e) **Includes**: permite *server-side* includes. Bastante usado em páginas dinâmicas;

f) **Indexes**: permite que, se determinada URL for solicitada e em sua pasta padrão (DocumentRoot) não houver um arquivo de index (como por exemplo: index.php, index.hml, index.asp), o diretório inteiro seja mostrado;

g) **Multiviews**: habilita múltiplas visões no servidor Web.

d) **AllowOverride**: diretiva que trata do controle de acesso ao servidor. Suporta os seguintes parâmetros:

a) **All**: habilita todas as funções de controle;

b) **None**: desabilita todas as funções de controle;

c) **AuthConfig**: permite o uso de diretivas de autorização, tais como: AuthUserFile, AuthType, AuthName, AuthGroupFile, etc.;

d) **FileInfo**: permite o uso de diretivas de controle de tipos de documentos, tais como: ErrorDocument, ForceType, SetHandler, etc;

e) **Indexes**: permite uso de diretivas de controle de indexação de diretórios, tais como: AddDescription, AddIcon, DefaultIcon, etc.;

f) **Limit**: permite uso de diretivas de controle de acesso ao *host*, tais como: Allow, Deny and Order.

e) **Order**: tem a função de controlar a ordem do controle de acesso ao servidor. Pode ser usado com três (3) tipos:

a) **Allow, Deny**: primeiramente, processa as diretivas Allow e, após isto, realiza a avaliação das diretivas Deny;

b) **Deny, Allow**: primeiramente, processa as diretivas Deny e, após isto, realiza a avaliação das diretivas Allow;

c) **Mutual-failure**: semelhante à cláusula "Allow, Deny", porém caiu em desuso.

Por fim, a cláusula "ordem agora" tem que definir quais serão as permissões de acesso, para isto podemos usar, por exemplo: "Allow from all". Isto permitirá que todos acessem o servidor. Por outro lado, se setarmos "Deny from all" o efeito é o contrário e ninguém conseguirá acessar o servidor. É possível, ainda, determinar nomes ou endereços IP na cláusula, por exemplo: "Allow from 192.168.1.1" ou "Deny from www.apache.org".

Com as configurações vistas nesta subseção, é possível estruturar o primeiro servidor Web. São configurações de certa forma básicas, mas que cobrem a maioria dos casos de uso de servidores Web. Mais informações sobre opções de controle, acesso e configuração do servidor Apache Web Server podem ser obtidas no *site* oficial da ferramenta.

RECAPITULANDO

Neste capítulo, você conheceu as teorias relacionadas a servidores Web. Estudou o histórico da tecnologia, que mostrou como Tim Berners-Lee idealizou o serviço e como passou de um simples sistema de controle de conteúdo à *World Wide Web*. Viu algumas características básicas que cobrem servidores Web, tais como *sites* virtuais e suporte a grandes arquivos. Após isto, foi visto como é estruturada a URL e quais são os componentes que formam uma URL.

Ainda neste capítulo, você viu uma introdução de segurança em servidores Web e o protocolo base que serve ao servidor Web, o OpenSSL, o qual possibilita ao servidor disponibilizar páginas seguras por meio do protocolo HTTPS. Viu, também, como é distribuído o mercado de servidores Web no mundo em uma pesquisa recente realizada pela NetCraft e, por fim, acompanhou uma instalação do servidor Apache Web Server, com um exemplo de *sites* virtuais e toda a explicação necessária pra compreender o exemplo.

Todas as informações apresentadas até aqui são fundamentais para a sua formação na área, mas, o aprendizado não para por aqui. Continue atento porque ainda tem muita coisa para aprender. Vamos em frente!

Anotações:

Servidor de *E-mail*

6

O tema deste capítulo são os servidores de *e-mail*. Durante o estudo, você passará pela história do serviço, suas principais características e definições (MTA, MUA, AA, LDA, etc), seus protocolos (SMTP, POP, IMAP, ESMTP, etc.) e por fim, pela instalação na prática de um servidor de *e-mail*. A compreensão dos conceitos básicos dos servidores e seus componentes é fundamental para que o administrador de sistemas possa estruturar e gerenciar o serviço de forma técnica e funcional.

Ao final deste capítulo, você terá subsídios para:

a) conhecer a evolução e a história dos servidores de *e-mail*;

b) entender os componentes dos sistemas de *e-mail*;

c) conhecer as aplicações para servidores de *e-mail*;

d) compreender o funcionamento do *webmail*;

e) conhecer uma lista de protocolos e portas relacionadas.

Mantenha seu passaporte em mãos para embarcar nas informações sobre os servidores de *e-mail*. Boa viagem!

6.1 HISTÓRIA

O correio eletrônico ainda é o serviço de usuário mais importante da Internet. A web carrega um volume maior de tráfego, mas o *e-mail* é o serviço usado para a maioria das comunicações de pessoa a pessoa. Nenhuma rede está completa sem o *e-mail* e nenhum sistema operacional de rede vale o seu preço se não incluir suporte a correio TCP/IP completo. (HUNT, 2004, p.141).

A importância deste tipo de servidor é notória e foi um avanço na comunicação global. A pergunta é simples: há quanto tempo você não recebe ou envia uma carta?

fotógrafo

Faz tempo,não é mesmo? E um *e-mail*? Quando foi que você recebeu ou enviou o último?

fotógrafo

A resposta para esta pergunta é uma evidência clara da necessidade que temos hoje em nos comunicar via *e-mail*. Atualmente, é impossível pensar em comunicação textual e não relacionar o *e-mail* como um dos principais componentes.

Esta premissa é verdadeira, porém alguns especialistas apontam que os IM (*Instant Messengers* ou mensageiros instantâneos) embutidos em sistemas Web, como Facebook, Gmail, Google Plus, etc., somados aos já conhecidos Skype, Microsoft MSN Messenger, AOL Messenger, etc., se tornarão mais importantes em pouco tempo, substituindo os sistemas de *e-mail*.

Com o advento do *Compatible Time-Sharing Sytem* (CTSS) ou Sistema Compatível de Compartilhamento de Tempo, em 1961, pela primeira vez, vários usuários foram capazes de acessar um sistema central para compartilhar e armazenar arquivos no disco do servidor. Com isso, métodos mais formais foram desenvolvidos para trocar mensagens e para criar os primeiros servidores de *e-mail*. (MAIL SERVER, 2011). Estes adventos foram:

a) 1965 - MIT's CTSS Mail System;

b) 1972 - Unix Mail Program;

c) 1972 - Mail Box APL, desenvolvido por Larry Breed;

d) 1981 - PROFS, desenvolvido pela IBM;

e) 1982 - ALL-IN-1, desenvolvido pela Digital Equipment Corporation.

Embora os conceitos sobre os primeiros sistemas de *e-mail* sejam parecidos, as características técnicas eram muito diferentes e não havia interoperabilidade entre os sistemas. Tais sistemas só permitiam a comunicação entre usuários que estivessem conectados ao mesmo *mainframe*. Com isto em mente, os desenvolvedores iniciaram esforços para desenvolver sistemas que fossem compatíveis entre diferentes organizações.

No ano de 1971, o primeiro *e-mail* foi enviado na ARPANET, usando as RFC's 561, 680 e 724 como padrão de comunicação e interoperabilidade. Em 1977, foi lançada a RFC 733, que tornou-se um padrão de troca de correio eletrônico na rede. Outros sistemas e padrões surgiram posteriormente, como: o "uucp", em 1978; BITNET, da IBM, em 1981; FidoNET, da IBM, em 1984; porém, estes não se tornaram padrão de uso e terminaram caindo em desuso.

VOCÊ SABIA?

Em 1972, ao desenvolver o primeiro programa de correio eletrônico (*e-mail*), Ray Tomlinson aproveitou o sentido "@" (at), disponível no teclado, e utilizou-o entre o nome do usuário e o nome do provedor. Assim "Fulano@ Provedor X" ficou significando "Fulano no provedor X". (ORIGEM DO @, 2011).

Você acompanhou uma breve história de como surgiu o *e-mail*, mas sabe de que componentes ele é formado? Não? Então continue acompanhando os próximos itens desse material e descubra isso e muito mais!

6.2 COMPONENTES DE UM SISTEMA DE *E-MAIL*

De acordo com Nemeth et al. (2007), um sistema de *e-mail* é formado por quatro componentes distintos, sendo eles:

a) **MUA** (*Mail User Agente*) ou o Agente de Usuário de Correio, que permite ao usuário ler e compor mensagens;

b) **MTA** (*Mail Transfer Agent*) ou Agente de Transporte de Correio, que tem a função de rotear e direcionar as mensagens entre as máquinas;

c) **LDA** (*Local Delivery Agent*) ou Agente de Entrega Local, que tem a função de colocar as mensagens em um armazenamento local de mensagens (caixas postais);

d) **AA** (*Access Agent*) ou Agente de Acesso, que conecta o agente de usuário ao local onde as mensagens estão armazenadas. Este componente é opcional na arquitetura dos sistemas de *e-mail*.

O agente de usuário de correio (MUA) é usado por usuários para ler e compor mensagens. No início dos sistemas de *e-mail*, as mensagens eram basicamente compostas por texto puro, mas um padrão conhecido como MIME (*Multipurpose Internet Mail Extesions*) é agora utilizado para codificar formatos de texto e anexar mensagens. Este componente é suportado pela grande maioria dos agentes de usuários (NEMETH et. al., 2007). No quadro a seguir, é possível visualizar exemplos de agentes de usuários de correio. Confira!

EXEMPLOS DE MUA	DESCRIÇÃO
mail	É o comando original dos UNIX e pode ser encontrado em uma grande variedade de clientes e servidores Linux.
Thunderbird	*Software* desenvolvido pela Mozilla que roda em Linux, Windows, Mac OS X, entre outros sistemas.
Pine	Foi desenvolvido pela Universidade de Washington e pode ser utilizado na linha de comando de sistemas UNIX ou Linux.
Eudora	Sofware desenvolvido pela Qualcomm, que pode ser usado em Mac OS X e Windows. Está caindo em desuso.
Outlook	*Software* desenvolvido pela Microsoft para leitura de *e-mail*. Roda somente em sistemas Windows. Há ainda uma versão melhorada que compõe o Microsoft Office que oferece uma grande quantidade de opções.
Mail	*Software* desenvolvido pela Apple para rodar em Mac OS X. Tem um poderoso sistema de integração entre calendários e agenda.

Quadro 5 - Exemplos de agentes de usuários de correio

O agente de transporte de correio (MTA) tem a função de receber um *e-mail* de um agente de usuário (MUA), compreender os endereços dos receptores e enviar o *e-mail* para o servidor de destino. A grande maioria dos agentes de transporte também atua como agente de envio de mensagens. Os agentes de transporte de correio "conversam" entre si por meio de um protocolo chamado *Simple Mail Transport Protocol* (SMTP) ou Protocolo de Transporte Simples de *E-mail*, definido na RFC 2821 ou através da Extend SMTP ou ESMTP e definido nas RFC's 1869, 1870 e 1985. (NEMETH et al., 2007).

Ainda, sobre os agentes de entrega, eles são de fato as aplicações que estruturam o serviço de *e-mail*. Entre estas aplicações podemos citar Microsoft Exchange, Postfix Mail System, Sendmail, Qmail, Exim, entre outros. Mais a frente, haverá um exemplo de instalação de um agente de entrega (MTA) utilizando o *Postfix Mail System* como base.

Os agentes de entrega locais (LDA) são os componentes que têm a função de receber as mensagens advindas do agente de transporte e entregá-las nas caixas de mensagens dos usuários. O aplicativo "mail" do UNIX é o agente de entrega padrão para usuários locais neste tipo de sistema. Outro tipo de agente de entrega é o *Procmail*. (PROCMAIL, 2011). Esse último é mais robusto e flexível que o *mail* e também é o agente de entrega do *Postfix*. (POSTFIX, 2011).

Os agentes de acesso (AA) são os componentes que proporcionam o acesso dos usuários às caixas de mensagens no servidor. Para isto, há dois protocolos que são amplamente utilizados, sendo eles: o *Internet Message Access Protocol* (IMAP) e o *Post Office Protocol* (POP). A característica básica do IMAP é que ele sincroniza as mensagens entre o MUA e o servidor de *e-mails*. Se você deletar uma mensagem no seu MUA, ela será apagada no servidor. Apesar desta forma ser mais pesada, é muito utilizada, porque as mensagens ficam no servidor. Sobre o POP, a característica básica é que ele faz o *download* das mensagens, desta forma, quando uma operação com o POP ocorre, ele apaga as mensagens do servidor, mantendo apenas uma cópia local.

Os componentes acima designam a arquitetura básica de sistemas de *e-mail*, porém, é importante salientar que estes não estão resumidos somente a eles. Outros componentes importantes podem compor uma arquitetura de sistemas de *e-mail*: filtros de mensagens, mecanismos *anti-spam*, mecanismos de anti-vírus, etc. Na figura a seguir, é possível visualizar uma comunicação simples na troca de *e-mails*.

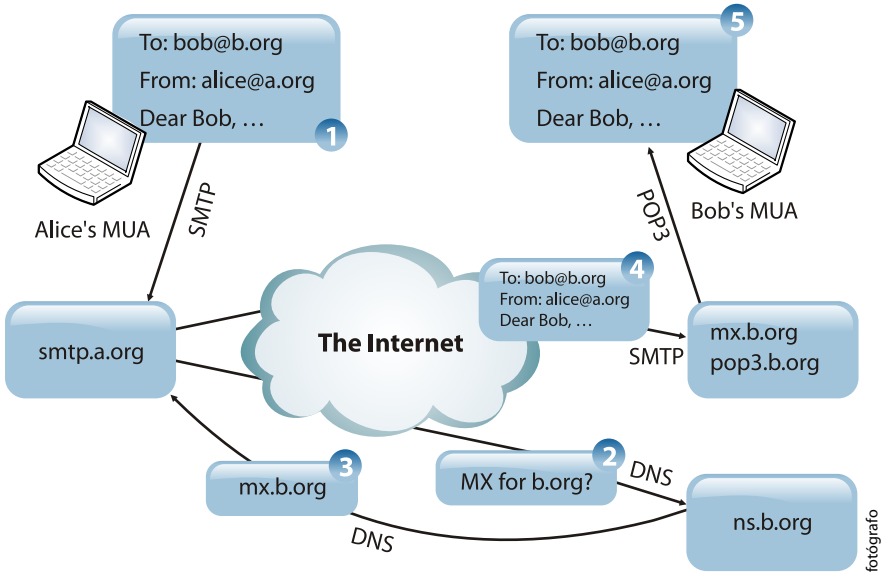


Figura 12 - Exemplo de arquitetura de troca de *e-mails*

Nessa figura, pode ser vista uma simulação de um *e-mail* enviado de Alice para Bob, por meio da Internet e é possível entender a operação que envolve o mesmo. Confira a seguir os detalhes da operação realizada.

a) O usuário Alice está editando uma mensagem em seu MUA para enviar para Bob. É importante salientar que Alice tem o *e-mail* "alice@a.org", que faz parte do domínio "a.org", cujo SMTP é "smtp.a.org". Quando ela pressiona o botão enviar em seu MUA, ocorre o passo 2.

b) No passo 2, o SMTP de Alice (smtp.a.org) irá se comunicar com o servidor DNS do destino e perguntará qual o endereço do MX (*Mail Exchanger*) do domínio "b.org".

c) O servidor DNS do domínio "b.org" responde ao SMTP de Alice que o endereço do servidor de *e-mails* (MX) do domínio de destino é "mx.b.org".

d) Sabendo-se do endereço do MX do domínio de destino, o SMTP de Alice envia o *e-mail* para o servidor de *e-mails* (mx.b.org) de Bob. O servidor de *e-mails* do destino recebe as mensagens e armazena na caixa postal de Bob.

e) Na etapa 5, o usuário Bob acessa seu MUA e pede, via protocolo POP3, para resgatar as mensagens em sua caixa, finalizando o processo.

Então, associou os passos de envio de *e-mail* quando você envia uma mensagem para alguém? Fácil de entender, não é mesmo? Agora, conheça outro tema interessante: as aplicações para os servidores de *e-mail*.

6.3 APLICAÇÕES PARA SERVIDORES DE *E-MAILS*

Segundo MX Survey (2011), em 2010 foi realizada uma pesquisa e 85% dos servidores de *e-mail* são divididos entre: Postfix, Exim, Sendmail e Microsoft Exchange Server. Estas pesquisas, em geral, são difíceis de serem inquestionáveis, mas norteiam quando se procura comparar servidores de *e-mail*. Para dar uma visão melhor sobre as possibilidades sobre ferramentas para servidores de *e-mails*, observe o quadro a seguir, que demonstra algumas alternativas.

SMTP	IMAP/POP	FILTROS
AfterLogic WebMail Lite Atmail	agorum core Open Source	
Axigen	Alt-N Technologies	
Bongo	Apache James	
Citadel	Axigen	
ContactOffice	Bongo	
CommuniGate Pro	Citadel	
Courier	CommuniGate Pro	
Eudora Internet Mail Server	ContactOffice	Alt-N Technologies
Exim	Courier Mail Server	ASSP
FirstClass	Cyrus IMAP server	Axigen
fmaild	DBMail	Bayesian filters
Gammadyne Mailer	Dovecot	Bogofilter
Gordano Messaging Suite	Eudora Internet Mail Server	DSPAM
HMailServer	FirstClass	Gordano Messaging Suite
IBM Lotus Domino	Gordano Messaging Suite	Hexamail Guard
Ipswitch IMail Server	HMailServer	Kerio Connect
James (Java Apache Mail Enterpri-se Server)	Ipswitch IMail Server	MagicMail
	Kerio Connect	MagicSpam
Kerio Connect	IBM Lotus Domino IMAP4 Server	MailChannels
MagicMail	MagicMail	MailScanner
Mailtraq	Mailtraq	Mailtraq
MDaemon Email Server for Win-dows	Meldware Communication Server	MIMEDefang
		Procmail
Mercury Mail Transport System	Mercury Mail Transport System	PureMessage
Meldware Communication Suite	Microsoft Exchange Server	SurfControl
MeTA1 (successor of the sendmail X project)	Microsoft Windows POP3 Service	SpamAssassin
	Novell GroupWise	WinGate
Microsoft Exchange Server	Novell NetMail	Vipul's Razor
MMDF	Open-Xchange	Webroot
Novell GroupWise	Oracle Beehive	
Novell NetMail	Synovel Collabsuite	
Openwave Systems	UW IMAP	
Open-Xchange	WinGate	
Oracle Beehive	XMail	
Oracle Communications Messa-ging Exchange Server	Zarafa	
	Zimbra	
Postfix		

qmail		
qpsmtpd		
Scalix		
Sendmail		
Smail		
SparkEngine		
SMTP Proxy		
Sun Java System		
Synovel Collabsuite		
WinGate		
XMail		
XMS Email Application Server		
Zarafa		
Zimbra		
ZMailer		

Quadro 6 - Ferramentas para servidores de *e-mail*s
Fonte: Adaptado de List of Mail Servers (2011)

Muito bem. Você já conheceu a história, os componentes e as aplicações de um servidor de *e-mail*. Agora é hora de aprender a instalar e configurar um. Então, mãos à obra!

6.4 INSTALANDO E CONFIGURANDO DE UM SERVIDOR DE *E-MAIL*

Com o intuito de demostrar uma instalação de um servidor de *e-mails* e colocar os conceitos vistos em prática, a partir de agora, você acompanhará a demonstração da instalação do servidor de *e-mails* Postfix sobre um sistema operacional Debian Linux/GNU, versão Squeeze. (POSTFIX, 2011). O Postfix roda em AIX, BSD, HP-UX, IRIX, Linux, MacOS X, Solaris, Tru64 UNIX, e em outros tipos de sistema operacionais baseados em UNIX. Sua instalação em um sistema operacional Linux baseado em Debian pode ser feita via gerenciador de pacotes "apt-get" ou ainda, acessando o *site* oficial, baixando o pacote e instalado-o manualmente. Veja:

```
root@server:/# apt-get install postfix
```

Dependendo do nível de configuração do sistema, o instalador fará algumas questões relativas às configurações que deseja para o servidor. Independente disto, nós podemos, a qualquer momento, reconfigurar o servidor via linha de comando, usando o comando "dpkg-reconfigure". As respostas às questões do configurador refletirão no principal arquivo de configuração do Postfix, chamado "main.cf", que fica localizado no "/etc/postfix". Este arquivo pode ser visualizado a seguir.

```
  root@server:/etc/postfix# cat main.cf
  # See /usr/share/postfix/main.cf.dist for a com-
mented, more complete version


  # Debian specific:  Specifying a file name will cause
the first
  # line of that file to be used as the name.  The
Debian default
  # is /etc/mailname.
  #myorigin = /etc/mailname

  smtpd_banner = $myhostname ESMTP $mail_name (Debi-
an/GNU)
  biff = no

  append_dot_mydomain = no

  # Uncomment the next line to generate "delayed
mail" warnings
  #delay_warning_time = 4h

  readme_directory = no

  # TLS parameters
  smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snake-
oil.pem
  smtpd_tls_key_file=/etc/ssl/private/ssl-cert-
snakeoil.key
  smtpd_use_tls=yes
  smtpd_tls_session_cache_database = btree:${data_
directory}/smtpd_scache
```

```
  smtp_tls_session_cache_database  =  btree:${data_
directory}/smtp_scache

  # See /usr/share/doc/postfix/TLS_README.gz in the
postfix-doc package for
  # information on enabling SSL in the smtp client.

  myhostname = server.exemplo.com
  alias_maps = hash:/etc/aliases
  alias_database = hash:/etc/aliases
  myorigin = /etc/mailname
  mydestination = server.exemplo.com, exemplo.com,
localhost
  relayhost =
  mynetworks = 127.0.0.0/8 192.168.1.0/24
  mailbox_command = procmail -a "$EXTENSION"
  mailbox_size_limit = 0
  recipient_delimiter = +
  inet_interfaces = all
  inet_protocols = ipv4
```

O arquivo "main.cf" pode suportar uma vasta quantidade de parâmetros e configurações para customizar o servidor de *e-mails*. A configuração realizada e vista é funcional e entregará *e-mails* normalmente, entretanto, é importante definirmos os parâmetros que este arquivo pode suportar para que o leitor possa decidir, por si só, as configurações que o seu servidor de *e-mails* trabalhará. Confira alguns desses parâmetros.

a) myhostname: é o nome do servidor na Internet. Geralmente se usa o FQDN (*Full Qualified Domain Name*), como por exemplo: mail.exemplo.com.

b) alias_maps: local onde são colocadas as listas de apelidos, dos servidores de *e-mail* para usuários, por exemplo. Tal lista pode estar localizada em um arquivo, como no exemplo "hash:/etc/aliases" ou ainda, em um repositório externo.

c) alias_database: semelhante ao parâmetro "alias_maps", com a diferença de que a base de dados de apelidos, neste caso, deve ser local.

d) myorigin: nome do domínio que aparecerá no endereço de *e-mail*. Este parâmetro suporta tanto nomes, como arquivos de configuração e variáveis internas, como por exemplo: "$myhostname".

e) mydestination: lista de domínios que serão entregues pelo agente de transporte (MTA).

f) mynetworks: é a lista de redes ou endereços IPs que tem mais privilégios. Suporta endereços de rede, IPs e arquivos de configuração. Em resumo, este parâmetro é uma lista de endereços confiáveis.

g) mailbox_command: é um parâmetro opcional que é usado para determinar um agente de entrega de mensagens. No caso do exemplo, está sendo usado o *procmail*. Este componente tem a função de entregar a mensagem na caixa de mensagens do usuário.

h) mailbox_size_limit: parâmetro para determinar o tamanho máximo de qualquer caixa de mensagens no sistema. É um tipo de mecanismo de cota de espaço para usuários. O atributo "0" para este parâmetro determina que a caixa de mensagem é ilimitada.

i) inet_interfaces: corresponde às interfaces locais da máquina que responderão pelo serviço de *e-mail*. Suporta o atributo "all" que significa que todas as interfaces da máquina estarão escutando conexões para o servidor de *e-mails* e ele poderá responder em todas.

j) inet_protocols: parâmetro que determina em que protocolos o servidor de *e-mails* irá trabalhar. Suporta ipv4, ipv6 e "all" para determinar que são ambos os tipos de IP.

Por fim, existe um infinidade de parâmetros possíveis para inserirmos nos arquivos de configuração de nosso servidor de *e-mails*.

SAIBA MAIS

Encontre mais informações de possíveis parâmetros para completar uma configuração mais avançada, acesse o *site* do *Postfix* e confira uma vasta quantidade de parâmetros que este arquivo pode suportar. Acesse: <www.postfix.org>.

Dessa forma, estando pronto o servidor de *e-mails*, é possível instalar o agente de acesso (AA) para podermos proporcionar aos usuários formas de resgatar as mensagens. Como você já viu, nós temos algumas possibilidades de *softwares* que se integram com o *Postfix* para proporcionar os protocolos de acesso IMAP e POP. Dentre os mais populares, podemos citar o "courier" e o "cyrus". A seguir, você pode ver a instalação do *courier*. Acompanhe.

```
  root@server:/# apt-get install courier-imap cou-
rier-pop
  Lendo listas de pacotes... Pronto
  Construindo árvore de dependências
  Lendo informação de estado... Pronto
  Os pacotes extra a seguir serão instalados:
      courier-authdaemon  courier-authlib  courier-
authlib-userdb courier-base expect libfam0 libltdl7
tcl8.5
  Pacotes sugeridos:
    courier-doc courier-imap-ssl courier-pop-ssl ex-
pectk fam tclreadline
  Os NOVOS pacotes a seguir serão instalados:
    courier-authdaemon courier-authlib courier-auth-
lib-userdb courier-base courier-imap courier-pop ex-
pect libfam0 libltdl7
    tcl8.5
  0 pacotes atualizados, 10 pacotes novos instala-
dos, 0 a serem removidos e 2 não atualizados.
  É preciso baixar 3230 kB de arquivos.
  Depois desta operação, 8593 kB adicionais de espa-
ço em disco serão usados.
  Você quer continuar [S/n]? S
  Obter:1       http://debian.pop-sc.rnp.br/debian/
squeeze/main libltdl7 amd64 2.2.6b-2 [296 kB]
  Obter:2       http://debian.pop-sc.rnp.br/debian/
squeeze/main tcl8.5 amd64 8.5.8-2 [1599 kB]
  Obter:3       http://debian.pop-sc.rnp.br/debian/
squeeze/main expect amd64 5.44.1.15-4 [250 kB]
  Obter:4       http://debian.pop-sc.rnp.br/debian/
squeeze/main  courier-authlib  amd64  0.63.0-3  [83,2
kB]
  Obter:5       http://debian.pop-sc.rnp.br/debian/
squeeze/main courier-authdaemon amd64 0.63.0-3 [8250
B]
  Obter:6       http://debian.pop-sc.rnp.br/debian/
squeeze/main  courier-authlib-userdb  amd64  0.63.0-3
[36,7 kB]
  Obter:7       http://debian.pop-sc.rnp.br/debian/
squeeze/main libfam0 amd64 2.7.0-17 [28,8 kB]
```

```
  Obter:8       http://debian.pop-sc.rnp.br/debian/
squeeze/main courier-base amd64 0.65.0-3 [245 kB]
  Obter:9       http://debian.pop-sc.rnp.br/debian/
squeeze/main courier-imap amd64 4.8.0-3 [622 kB]
  Obter:10      http://debian.pop-sc.rnp.br/debian/
squeeze/main courier-pop amd64 0.65.0-3 [61,3 kB]
  Baixados 3230 kB em 0s (12,7 MB/s)
  Pré-configurando pacotes ...
  Selecionando pacote previamente não selecionado
libltdl7.
  (Lendo banco de dados ... 25411 ficheiros e direc-
tórios actualmente instalados.)
  Desempacotando libltdl7 (de .../libltdl7_2.2.6b-2_
amd64.deb) ...
  Selecionando pacote previamente não selecionado
tcl8.5.
  Desempacotando  tcl8.5  (de  .../tcl8.5_8.5.8-2_
amd64.deb) ...
  Selecionando pacote previamente não selecionado
expect.
  Desempacotando expect (de .../expect_5.44.1.15-4_
amd64.deb) ...
  Selecionando pacote previamente não selecionado
courier-authlib.
  Desempacotando courier-authlib (de .../courier-
authlib_0.63.0-3_amd64.deb) ...
  Selecionando pacote previamente não selecionado
courier-authdaemon.
  Desempacotando courier-authdaemon (de .../couri-
er-authdaemon_0.63.0-3_amd64.deb) ...
  Selecionando pacote previamente não selecionado
courier-authlib-userdb.
  Desempacotando  courier-authlib-userdb  (de  .../
courier-authlib-userdb_0.63.0-3_amd64.deb) ...
  Selecionando pacote previamente não selecionado
libfam0.
  Desempacotando libfam0 (de .../libfam0_2.7.0-17_
amd64.deb) ...
  Selecionando pacote previamente não selecionado
courier-base.
```

```
  Desempacotando courier-base (de .../courier-ba-
se_0.65.0-3_amd64.deb) ...
  Selecionando pacote previamente não selecionado
courier-imap.
  Desempacotando courier-imap (de .../courier-
imap_4.8.0-3_amd64.deb) ...
  Selecionando pacote previamente não selecionado
courier-pop.
  Desempacotando courier-pop (de .../courier-
pop_0.65.0-3_amd64.deb) ...
  Processando gatilhos para man-db ...
  Configurando libltdl7 (2.2.6b-2) ...
  Configurando tcl8.5 (8.5.8-2) ...
  update-alternatives: a usar /usr/bin/tclsh8.5 para
disponibilizar /usr/bin/tclsh (tclsh) em modo auto-
mático.
  Configurando expect (5.44.1.15-4) ...
  Configurando courier-authlib (0.63.0-3) ...
  Configurando courier-authdaemon (0.63.0-3) ...
  Starting Courier authentication services: authdae-
mond.
  Configurando courier-authlib-userdb (0.63.0-3) ...
  Configurando libfam0 (2.7.0-17) ...
  Configurando courier-base (0.65.0-3) ...
  update-alternatives: a usar /usr/bin/deliverquo-
ta.courier para disponibilizar /usr/bin/deliverquota
(deliverquota) em modo automático.
  update-alternatives: a usar /usr/share/man/man5/
maildir.courier.5.gz para disponibilizar /usr/share/
man/man5/maildir.5.gz (maildir.5.gz) em modo automá-
tico.
  update-alternatives: a usar /usr/bin/maildirmake.
courier para disponibilizar /usr/bin/maildirmake
(maildirmake) em modo automático.
  update-alternatives: a usar /usr/share/man/man7/
maildirquota.courier.7.gz para disponibilizar /usr/
share/man/man7/maildirquota.7.gz (maildirquota.7.gz)
em modo automático.
```

```
  update-alternatives: a usar /usr/bin/makedat.cou-
rier para disponibilizar /usr/bin/makedat (makedat)
em modo automático.
  Configurando courier-imap (4.8.0-3) ...
  Starting Courier IMAP server: imapd.
  Configurando courier-pop (0.65.0-3) ...
  Starting Courier POP3 server: pop3d.
```

FIQUE ALERTA

Tome muito cuidado com a configuração registro MX no servidor de nomes (DNS). Se ele estiver mal configurado, a comunicação de mensagens será prejudicada, causando a não entrega de mensagens.

Feitos os procedimentos de instalação do servidor de *e-mails* Postfix (MTA), juntamente com os agentes de acesso representados pelos *softwares* "courier--imap" (IMAP) e "courier-pop" (POP), agora nós temos a infraestrutura de *e-mails* pronta para ser utilizada. A próxima etapa é escolher um programa (MUA) pra interagir com sua caixa de mensagens no servidor, entre estes, podemos citar: Mail, Thunderbird, Outlook, entre outros.

Acompanhe o casos e relatos a seguir e confira um exemplo prático sobre o assunto tratado neste capítulo.

CASOS E RELATOS

Espionagem via servidores de *e-mail*

Uma empresa de construção civil constatou que, por três vezes, houve vazamento de informações sobre licitações da empresa, pois foram perdidas três licitações por menos de 2% do valor para a ganhadora, causando-lhe um prejuízo econômico de milhões. Sendo assim, a empresa contratou uma empresa de segurança de redes de computadores para fazer uma consultoria em todos os ambientes e checar o que estava havendo. Depois de três semanas de análise, a equipe de segurança solicitou uma reunião para discutir as falhas encontradas que culminaram no vazamento das informações sigilosas.

Toda vez que o orçamento de uma nova obra era definido, a empresa enviava os dados financeiros, via *e-mail*, para a contabilidade conferir os dados e cuidar da parte fiscal. O que ninguém imaginava é que um *cracker* havia sido contratado por uma empresa concorrente para espionar as comunicações dessa empresa e ele descobriu que não havia nenhuma camada de criptografia para proteger os conteúdos das mensagens de *e-mails*. Dessa forma, ficou fácil para o *cracker* interceptar as mensagens e remontar a informação. Para resolver o problema, a equipe de segurança "blindou" o servidor de *e-mails* com diversos tipos de autenticação, criptografia no SMTP, IMAP e POP, e o *webmail* da empresa passou a ter um certificado SSL. Também foi estruturada uma entidade certificadora (CA) que passou a criptografar com uma chave de 2048 bits usando o GPG, que é considerado, atualmente, inquebrável.

Viu só como é importante criptografar as mensagens de *e-mail*? É um processo simples que pode evitar muitos problemas, ou mesmo prejuízos. No próximo item, confira outra maneira para acessar a caixa de mensagem.

6.5 *WEBMAIL*

Outra forma de acesso à caixa de mensagens dos usuários é por meio da utilização de *softwares* de *webmail* para a interação. Estes *softwares* são considerados em sua essência como MUAs, já que exercem a mesma função, com a diferença de que eles proporcionam as interfaces de interação via Web. O modelo de operação destes *softwares* é baseado em servidores web, onde o *software* em si é caracterizado por uma aplicação que será disponibilizada em determinado endereço.

Para sistemas Linux, existem várias opções de *Webmail* para integrarmos com o nosso servidor de *e-mail*. Elas se diferenciam em características técnicas, como por exemplo, autenticação, suporte a calendário, agenda, etc. Desta forma, o administrador do sistema deve adequar as suas necessidades à melhor ferramenta para atendê-las. Entre as ferramentas existentes, podemos listar:

a) Squirrelmail;

b) Horde/IMP;

c) Zimbra;

d) Uebimiau;

e) OWA – *Outlook Web Acess*.

Como exemplo de instalação de uma das alternativas, podemos citar a instalação do *Squirrelmail*. O *software* é escrito na linguagem de programação web PHP e deve ser integrado com o servidor web da organização. Ele é muito simples e fácil de configurar. Veja o código da instalação:

```
root@server:/# apt-get install squirrelmail
```

Neste momento, a instalação é feita nas dependências do pacote e integrará a ferramenta ao servidor web instalado na máquina. É uma tarefa simples de ser realizada e a qualquer momento, se o administrador de sistemas quiser alterar as configurações do *webmail*, basta acessar a aplicação "conf.pl", que é um script que auxilia na configuração do *webmail*.

É importante salientar que a grande maioria dos *webmails* trabalha com o agente de acesso IMAP, visto que o POP não trabalha muito bem com a sincronização de mensagens. Na figura a seguir, é possível ver um exemplo de tela inicial do *webmail Squirrelmail*.

fotógrafo

Figura 13 - Exemplo de tela incial do *webmail Squirrelmail*

Para ajudar a identificar os protocolos e as portas que estão relacionadas aos protocolos nas configurações, confira a tabela a seguir.

Tabela 2 - Lista de protocolos e portas

PROTOCOLO	PORTA
SMTP	25
IMAP	143
POP3	110
Secure IMAP (IMAP4-SSL)	585
Secure POP3 (SSL)	995
Secure SMTP (SSMTP)	465
IMAP4 sobre SSL	993

RECAPITULANDO

Neste capítulo, você conferiu aspectos relacionados ao tema de servidores de *e-mail*. Foi visto que a história dos servidores de *e-mail* se confunde com a história da Internet em si e que os *e-mails* são a forma de comunicação textual mais usada na Internet. Você também aprendeu sobre o modelo de operação que este tipo de servidor exige, seus componentes e como eles se comunicam. Conferiu uma vasta lista de aplicações para construção de sistemas de *e-mail*, relacionando os MTA, MUA, filtros e afins. Acompanhou o exemplo de uma instalação utilizando o sistema Postfix como SMTP e o *Courier* servidor POP e IMAP para iteração com as caixas de mensagem. Por fim, conheceu características de webmails e as funções que eles podem exercer na arquitetura dos sistemas de *e-mail*. O próximo assunto que você estudará é sobre os "servidores de arquivos". Já ouviu falar deles? Sabe para que servem? Continue atento que logo você descobrirá!

Servidor de Arquivos

7

Neste capítulo, você verá os aspectos relacionados a servidores de arquivos. Conhecerá, por meio dos tipos de servidores de arquivos, o protocolo SMB/CIFS e uma abordagem para sistemas Linux que utiliza "Samba" e também uma demonstração de compartilhamento de arquivos no Windows. Ainda neste capítulo, você verá os sistemas de cotas que podem ser utilizados no Windows e no Linux, usando o sistema "quota". Por fim, acompanhará a demonstração do *Network File System* (NFS) em sistemas UNIX e um exemplo de instalação para este tipo de servidor de arquivos.

Ao final deste capítulo, você terá subsídios para:

a) conhecer os tipos de servidores de arquivos;

b) entender os protocolos SMB/CIFS;

c) compreender o funcionamento dos serviços NFS;

d) conhecer implementações de quotas em sistemas operacionais.

7.1 TIPOS DE SERVIDORES DE ARQUIVOS

Os servidores de arquivos, ou *file servers*, são por definição, computadores que estão anexados a uma rede de computadores e têm por função disponibilizar meios de acesso à escrita e leitura de arquivos, como documentos, planilhas, músicas, filmes, imagens, banco de dados, etc., que estão armazenados em discos compartilhados remotamente.

Este tipo de tecnologia surgiu quando houve a necessidade de tirarmos das estações de trabalho a responsabilidade de efetuar *backups* de todos os arquivos. Esse tipo de abordagem surgiu, também, para facilitar a distribuição de arquivos, visto que distribuir grandes quantidades de dados via outros protocolos não é tão trivial e prático. Desta forma, essa tecnologia ganhou espaço e está presente praticamente em todas as organizações.

Os servidores de arquivos podem ser classificados em dois tipos: dedicados e não-dedicados. Um servidor de arquivos dedicado é desenhado para um único propósito: servir arquivos. Ele possui todas as características e facilidades requeridas para servidores desta natureza, tais como controle de acesso baseados em usuários e senhas, controle de cota de usuários, compartilhamento de múltiplos espaços, autorizações baseadas em grupos de trabalho, etc.

Este tipo de componente de rede ainda pode ser categorizado pelos meios de acesso que ele proporciona, tais como os *Internet File Server*, que em sua maioria são acessados via protocolo FTP (*File Transfer Protocol*) ou por HTTP (*Hyper-Text Transfer Protocol*). Hoje em dia, ainda é possível compartilhar arquivos nesta categoria via protocolos modernos, tais como o *torrent* ou outros protocolos P2P existentes. O modo de operação pode ser visto na figura a seguir.

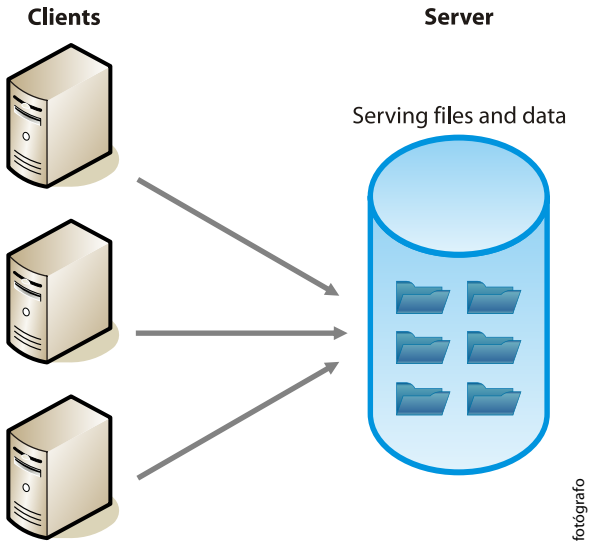


Figura 14 - Modelo de operação dos servidores de arquivos

Sobre os *Network File Server* são aqueles que ficam resididos dentro das redes locais das empresas e organizações e podem ser acessados via protocolo SMB/CIFS, NFS (*Network File Server*), servidores FTP locais e, ainda, servidores HTTP locais. Então, basicamente os protocolos servem somente como meio de acesso, podendo ser usados para qualquer um dos fins, desde que se atinjam os requisitos exigidos.

Os protocolos e servidores de arquivos abordados neste capítulo serão o SMB/CIFS (por trabalhar com Windows e Linux) e o NFS. Essa definição se dá pelo fato de que no decorrer da unidade, os protocolos HTTP e FTP serão estudados, desta forma, seria redundante manter dois locais com o mesmo conteúdo.

7.2 SMB/CIFS

O *Server Messabe Block* (SMB) ou também conhecido como *Common Internet File System* (CIFS) é um protocolo que trabalha na camada de aplicação do TCP/IP e tem como principal função fornecer acesso compartilhado a arquivos, impressoras, portas seriais e vários outros tipos de comunição em redes de computadores. Ele trabalha nas portas 137 e 138 (UDP) e portas 137 e 139 (TCP).

Por volta dos anos 1990, Barry Feigenbaum desenvolveu o SMB para os sistemas DOS, para acesso de arquivos locais por meio da rede. A Microsoft fez modificações consideráveis para adaptar a versão do protocolo para seus sistemas operacionais e ainda fundiu o SMB com o conhecido produto, chamado LAN Manager. Por volta de 1992, a Microsoft continuou fazendo melhorias e surgiu o *Windows for Workgroups*, que focava seus esforços para ser a plataforma de troca de arquivos para grupos de trabalhos, facilitando o trabalho colaborativo.

Em 1996, a Microsoft lançou uma iniciativa para trocar o nome do SMB para *Common Internet File System* (CIFS), incrementando, desta forma, o novo SMB com várias novas funcionalidades como suporte a *links* simbólicos, tamanhos de arquivos maiores e *links* para discos rígidos. Tudo isso em uma tentativa de suportar conexões diretas sobre a 445 (TCP) sem a necessidade do NetBIOS, pois o Server Message Block foi projetado para trabalhar sobre o NetBIOS/NetBEUI desde a sua concepção. Sendo assim, desde o Windows 2000 Server, o CIFS é executado, por padrão, na porta 445 (TCP) em vez da porta 139 (TCP). Em 2006, a Microsoft inseriu a segunda versão do SMB no Windows Vista e no Windows 7, sendo que agora ela é padrão para sistemas Windows. (CIFS, 2011).

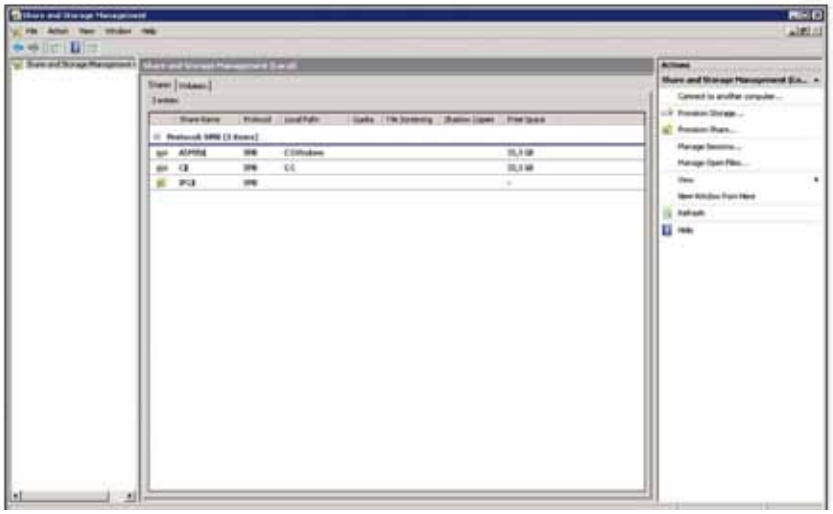
Em 1992, um estudante de doutorado da Universidade Nacional da Austrália chamado Andrew Tridgell desenvolveu um sistema que implementava o protocolo SMB e o chamou de "Samba". O Samba foi desenvolvido usando métodos de engenharia reversa já que a Microsoft não disponibiliza as definições deste protocolo proprietário. O objetivo de Andrew era ter um sistema que conseguisse interoperabilidade com compartilhamentos de arquivos e impressoras entre Windows e Linux.

Atualmente, como você já viu, temos implementações do SMB/CIFS da plataforma Windows para clientes que compartilham pastas até servidores de arquivos, baseados em sua família de Servers. Já para sistemas operacionais baseados em UNIX, tais como Linux, Mac OS X, AIX e HP-UX existe o Samba, que implementa o procolo SMB. Nas próximas etapas deste material, você verá como é possível implementar este protocolo em ambas plataformas.

7.2.1 WINDOWS

Os sistemas operacionais da família Windows já vêm com todo suporte para conexão e compartilhamento (cliente e servidor) de arquivos. Mesmo as versões para estações de trabalho como Windows 7, Windows Vista e XP, já vêm com suporte a compartilhamento de pastas, entretanto, não é tão sofisticado como nos sistemas operacionais da família de servidores Windows, que tem suporte a múltiplas autenticações, política de cotas (como veremos a frente), entre outras características.

Para acessar a parte de Gerenciamento de Armazenamento e Compartilhamento do Windows 2008 Server (Datacenter) você deve ir em "Iniciar", depois "Ferramentas Administrativas" e após, em "Gerenciamento de Armazenamento e Compartilhamento". Nesta aplicação é possível fazer as definições de compartilhamento que achar necessário para atender os requisitos exigidos. É uma configuração bem simplificada na qual, com apenas alguns cliques, já estará tudo funcionando e convergido com as outras políticas.



fotógrafo

Figura 15 - Central de Gerenciamento de Compartilhamento do Windows 2008 Server Datacenter

7.2.2 SAMBA

Você já sabe que no sistema operacional Linux, o pacote que disponibiliza e implementa o protocolo SMB/CIFS mais popular é o Samba. Ele pode ser encontrado em uma gama de sistemas operacionais baseados em UNIX e hoje é um padrão quando há a necessidade de compartilhamento de arquivos entre sistema UNIX-like e Windows-like. A seguir, você pode ver um exemplo de instalação do servidor de arquivos Samba, em um sistema operacional Debian GNU/Linux na versão *Squeeze*. Confira!

```
root@molar:/home/douglas# apt-get install samba
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  samba-common samba-common-bin
Suggested packages:
   openbsd-inetd inet-superserver smbldap-tools ldb-
tools ctdb
```

```
  The following NEW packages will be installed:
    samba samba-common samba-common-bin
  0 upgraded, 3 newly installed, 0 to remove and 26 not
upgraded.
  Need to get 15.1 MB of archives.
  After this operation, 43.4 MB of additional disk space
will be used.
  Do you want to continue [Y/n]? Y
  Get:1 http://security.debian.org/ squeeze/updates/main
samba-common all 2:3.5.6~dfsg-3squeeze5 [388 kB]
  Get:2 http://security.debian.org/ squeeze/updates/main
samba amd64 2:3.5.6~dfsg-3squeeze5 [8,334 kB]
  Get:3 http://security.debian.org/ squeeze/updates/main
samba-common-bin amd64 2:3.5.6~dfsg-3squeeze5 [6,378 kB]
  Fetched 15.1 MB in 6s (2,460 kB/s)
  Preconfiguring packages ...
  Selecting previously deselected package samba-common.
  (Reading database ... 195215 files and directories cur-
rently installed.)
  Unpacking      samba-common      (from      .../samba-
common_2%3a3.5.6~dfsg-3squeeze5_all.deb) ...
  Selecting previously deselected package samba.
  Unpacking   samba   (from   .../samba_2%3a3.5.6~dfsg-
3squeeze5_amd64.deb) ...
  Selecting previously deselected package samba-common-
bin.
  Unpacking  samba-common-bin  (from  .../samba-common-
bin_2%3a3.5.6~dfsg-3squeeze5_amd64.deb) ...
  Processing triggers for man-db ...
  Setting up samba-common (2:3.5.6~dfsg-3squeeze5) ...

  Creating config file /etc/samba/smb.conf with new version
  Setting up samba (2:3.5.6~dfsg-3squeeze5) ...
  Generating /etc/default/samba...
  tdbsam_open: Converting version 0.0 database to version
4.0.
  tdbsam_convert_backup:  updated  /var/lib/samba/passdb.
tdb file.
```

```
Importing account for nobody...ok
Importing account for douglas...ok
Importing account for alexandre...ok
Adding group `sambashare' (GID 127) ...
Done.
```

O arquivo de configuração principal do Samba é o "/etc/samba/smb.conf". Isto varia de distribuição para distribuição mas, em sua maioria, fica localizado no diretório "/etc". Neste arquivo concentram-se todas as informações relativas à configuração do servidor de arquivos, podendo configurar diretórios, configurações de autenticação, compartilhamento de impressoras, entre outros aspectos.

VOCÊ SABIA?

É possível monitorar o servidor de arquivos para não permitir gravações de arquivos de áudio e vídeo usando o servidor de arquivos Samba. Para isto, use a cláusula "veto files".

Este arquivo é dividido em seções, representadas como "[NOME DA SESSAO]", seguido dos parâmetros. Por padrão, o Samba reserva algumas seções para definição de diretórios, impressoras e configurações globais. Veja, no quadro a seguir, as quatro seções e suas definições.

SEÇÕES	DESCRIÇÃO
[global]	Todas as configurações que estão nesta seção afetam o servidor como um todo, como por exemplo, o modo de autenticação dos usuários, definição de domínios e grupos de trabalho, *hostname* do computador, entre outros.
[homes]	Define as configurações dos homes dos usuários. Os *homes* dos usuários geralmente ficam dispostos em "/home".
[printers]	Define as configurações relativas ao compartilhamento de impressoras pelo Samba.
[profile]	Este parâmetro define os perfis quando utilizada a função de controlador de domínio (PDC) do Samba. Isto é usado quando máquinas Windows utilizam o Samba como PDC.

Quadro 7 - Sessões de configuração do Samba Server

A seguir, você pode visualizar o trecho do arquivo de configuração “smb.conf” que trata o compartilhamento dos “homes” (/home/*) dos usuários, um diretório chamado “public” (/home/public) que está aberto para todos os usuários e um diretório chamado “dados” (/home/dados). Todos estes diretórios têm configurações de acesso, leitura e gravação. É muito importante salientar que as permissões das pastas no sistemas operacional afetam a configuração do Samba. Desta forma, não adianta o parâmetro de permissões estar 0777 se no sistema operacional está 0700. Sendo assim, somente o dono do diretório gravará arquivos no mesmo.

```
[homes]
  comment = Home Directories
  browseable = no
  read only = yes
  create mask = 0700
  directory mask = 0700
  valid users = %S

[public]
  path = /home/public
  guest ok = yes
  browseable = yes
  writeable = yes
  printable = no
  create mask = 0777
  force create mode = 0777

[dados]
  path = /home/dados
  guest ok = no
  browseable = yes
  read only = no
  writeable = yes
  create mask = 0755
  force create mode = 0755
```

SAIBA MAIS

Para você saber mais sobre a configuração do servidor de arquivos Samba *Server*, acesse o *site* oficial da ferramenta: <http://www.samba.org>.

Vários parâmetros podem ser definidos para configuração das seções no Samba. Confira a lista dos parâmetros mais utilizados neste servidor de arquivos.

a) **comment**: apenas um comentário do compartilhamento para identificação.

b) **path**: este é o caminho no sistema operacional do diretório que será compartilhado pelo Samba.

c) **writeable**: isto indica se é permitido criar ou excluir arquivos ou diretórios no compartilhamento.

d) **public**: parâmetro que define se será permitido o acesso por outros usuários que não sejam os definidos.

e) **browseable**: parâmetro que define se o diretório será exibido ou não no ambiente de compartilhamento de redes dos sistemas operacionais.

f) **write list**: este parâmetro define se os usuários ou grupos relacionados têm acesso à escrita no compartilhamento.

g) **read list**: este parâmetro define que os usuários listados aqui têm permissão de leitura no compartilhamento.

h) **force create mode**: este parâmetro dita ao servidor de arquivos que é para o tipo de permissão criada dentro do compartilhamento. Se setado para 0755, a máscara do arquivo ou pasta nova no sistema será 0755.

Acompanhe um exemplo desse assunto no Casos e relatos a seguir.

CASOS E RELATOS

Servidores de arquivos para estudantes

Em uma instituição de ensino superior, a equipe de TI vinha sofrendo com o espaço em disco utilizado pelos alunos nos sistemas de armazenamento nos servidores de arquivos e nos servidores de *e-mail*. Não havia política de cotas definidas quando o sistema surgiu, desta forma, o sistema não foi projetado para crescer tanto.

Na época que o servidor de arquivos foi criado, a faculdade tinha apenas 300 alunos, o que não era um problema gerencial tão grande, porém, quatro anos depois, a faculdade já estava com 1200 alunos. A solução encontrada foi definir um servidor de arquivos construído em um sistema operacional Linux, usando o Samba e o sistema de cotas (quota). Foi definida uma cota de 100Mb por aluno, para serem usados tanto via servidor de arquivos como servidor de *e-mails*. Dessa forma, foi necessário estruturar um servidor com poder de armazenamento na ordem de 4 terabytes, com redundância feita em RAID 5, já preparando o ambiente para um crescimento esperado nos próximos anos.

Legal esse exemplo, não é mesmo? Fácil de entender. No próximo item, confira informações sobre o *Network File System*, ou seja, o NFS.

7.3 NFS

O *Network File System* (NFS) foi desenvolvido, em 1984, pela *Sun Microsystems* para permitir que clientes remotos compartilhem arquivos na rede. É um protocolo maduro definido na RFC 1094 na versão 1. Com o desenvolvimento do protocolo, ele foi redefinido em 1988, trabalhando sobre UDP, e em dezembro de 2000, o NFSv4 foi lançado e definido na RFC 3530, incluindo melhoramentos no desempenho na transferência dos arquivos.

Ele funciona basicamente em todas as variantes de UNIX, incluindo Linux, AIX, HP-UX, Mac OS X, FreeBSD, OpenBSD, entre outros. Como exemplo, vamos fazer uma instalação de um servidor de arquivos NFS em um sistema operacional Debian GNU/Linux na versão *Squeeze*.

Veja um exemplo usando o sistema gerenciador de pacotes APT.

```
  root@server:/etc# apt-get install nfs-kernel-serv-
er
  Lendo listas de pacotes... Pronto
  Construindo árvore de dependências
  Lendo informação de estado... Pronto
  Os NOVOS pacotes a seguir serão instalados:
```

```
  0 pacotes atualizados, 1 pacotes novos instalados,
0 a serem removidos e 2 não atualizados.
  É preciso baixar 170 kB de arquivos.
  Depois desta operação, 422 kB adicionais de espaço
em disco serão usados.
  Obter:1       http://debian.pop-sc.rnp.br/debian/
squeeze/main nfs-kernel-server amd64 1:1.2.2-4 [170
kB]
  Baixados 170 kB em 0s (1009 kB/s)
  Selecionando pacote previamente não selecionado
nfs-kernel-server.
  (Lendo banco de dados ... 34449 ficheiros e direc-
tórios actualmente instalados.)
  Desempacotando nfs-kernel-server (de .../nfs-ker-
nel-server_1%3a1.2.2-4_amd64.deb) ...
  Processando gatilhos para man-db ...
  Configurando nfs-kernel-server (1:1.2.2-4) ...

  Creating config file /etc/exports with new version

  Creating config file /etc/default/nfs-kernel-server
with new version
  Starting NFS common utilities: statd.
  Exporting directories for NFS kernel daemon....
  Starting NFS kernel daemon: nfsd mountd.
```

Depois de instalado o pacote, o servidor já tem suporte para compartilhar arquivos via rede. O arquivo principal de configuração do servidor NFS é o "/etc/exports". Nele, é possível compartilhar pastas de uma maneira bem flexível, baseadas em rede ou IP, com opções de controle sobre o conteúdo, como podemos ver no exemplo a seguir.

```
  # Diretório        Rede/IP dos clientes       Per-
missões de acesso
  /home/administrativo  192.168.1.0/24      (rw,no_
root_squash,sync)
  /home/financeiro      192.168.2.0/24      (ro,no_
root_squash,sync)
```

Como você pôde ver, é bem simples configurar um servidor NFS para compartilharmos pastas no sistema. Basicamente, ele segue o formato: diretório, IP ou rede que irá acessar o diretório compartilhado e as permissões de acesso à pasta.

Sobre as permissões de acesso, no quadro a seguir, confira uma lista das opções e suas descrições.

PERMISSÃO	DESCRIÇÃO
ro	Exporta somente leitura.
rw	Exporta para leitura e gravação.
root_squash	Mapeia UID e GID 0 para os valores especificados por anonuid e anongid.
no_root_squash	Permite acesso normal por parte de *root*.
all_squash	Associa todos os UIDs e GIDs às suas versões anônimas, é útil para suportar PCs.
secure	Requer que o acesso remoto se origine de uma porta privilegiada.
insecure	Permite acesso remoto de qualquer porta.
noaccess	Impede o acesso a esse diretório e seus subdiretórios.
async	Faz o servidor responder a requisições de gravação antes de uma gravação no disco real. Modo assíncrono.
sync	Funciona de forma síncrona, ou seja, o servidor responde à gravação imediatamente no disco real.

Quadro 8 - Algumas opções de acesso do NFS

Por fim, após o compartihamento e configuração dos acessos, o cliente pode acessar o servidor NFS onde os arquivos estão compartilhados. Isto pode ser feito, manualmente, usando o comando “mount”, ou ainda, pode ser feito usando o arquivo “fstab”. Dessa forma, toda vez que o sistema inicializar, a partição “subirá” juntamente com a máquina cliente. Veja um exemplo de um arquivo “/etc/fstab” configurado para acessar um servidor de arquivos NFS na inicialização.

```
root@server:/etc# cat /etc/fstab
# /etc/fstab: static file system information.
#
# <file system> <mount point>     <type>    <options>
<dump>  <pass>
proc            /proc           proc    defaults        0
0
# / was on /dev/sda1 during installation
UUID=a7778313-236a-4eef-a6f1-2c10de9605a1 /        ext3
errors=remount-ro 0       1
/dev/scd0       /media/cdrom0   udf,iso9660 user,noauto
0       0
/dev/fd0        /media/floppy0  auto    rw,user,noauto
0       0
# NFS
192.168.1.1:/home/financeiro /media/financeiro    nfs  rw
0 0
```

Por fim, é possível montar a diretório compartilhado manualmente, no tempo que quiser. Para isto, é usado o comando "mount", como pode ser visto a seguir. O formato do comando é dado provendo o endereço ou nome do servidor de arquivos, o compartilhamento e o ponto onde este conteúdo será montado.

```
# mount –t nfs 192.168.1.1:/home/administrativo /mnt
```

O *Network File Server* (NFS) ainda é muito utilizado, principalmente para operações pequenas de compartilhamento de arquivos que não exijam níveis mais sofisticados de gerência, como o controle de acesso baseado em usuários e grupos, o controle sobre conteúdo e a interoperalidade nativa com sistemas operacionais Windows-like. Ele é muito utilizado em infraestruturas para realização de *backup* (cópia de segurança) entre os servidores.

Você acompanhou a configuração do NFS e já acompanhou algumas informações sobre as cotas, mas sabe o que são essas cotas? Confira no item a seguir.

7.4 COTAS

Um assunto muito comum no âmbito dos servidores de arquivos é o sistema de quotas que será utilizado. Cotas podem ser definidas como uma quantidade de informação que cada usuário ou grupo pode armazenar nos servidores de rede. É importante salientar que a política de cotas (quotas) pode ser utilizada para vários aspectos dos serviços de redes, entre eles, servidores de redes, servidores de *e-mail*, servidores Web, entre outros.

Em sistemas baseados em Windows, esta funcionalidade está presente na família dos *Servers* (2003 e 2008) e pode ser acessada pela ferramenta de gerenciamento de discos. Essa funcionalidade proporciona a limitação do espaço que será utilizado pelos usuários para a gravação de seus arquivos em disco. Nela podem ser criadas cotas individuais de tamanhos variados, independente da hierarquia do usuário no sistema operacional.

Na figura a seguir, você pode ver a tela de configuração no Windows 2003 Server.

fotógrafo

Figura 16 - Ferramenta de configuração de cotas no Windows

FIQUE ALERTA

É muito importante realizar um estudo de implantação de cotas antes de sua implantação de fato, pois, fazer ajustes na política com o servidor já em produção pode ser bastante custoso. Então, fique alerta!

Nos sistemas operacionais baseados em Unix, tais como Linux, AIX ou Mac OS X, quando um administrador quer usar um sistema de cotas, ele geralmente usa a ferramenta "quota", que está disponível para qualquer sistema desta natureza.

Veja a instalação da ferramenta em um sistema operacional Debian GNU/Linux na versão *Squeeze*.

```
root@server:/home/douglas# apt-get install quota
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  libnet-ldap-perl
The following NEW packages will be installed:
  quota
0 upgraded, 1 newly installed, 0 to remove and 26 not
upgraded.
Need to get 580 kB of archives.
After this operation, 1,610 kB of additional disk space
will be used.
Get:1  ftp://ftp.pucpr.br/debian/  squeeze/main  quota
amd64 4.00~pre1-6 [580 kB]
Fetched 580 kB in 0s (635 kB/s)
Preconfiguring packages ...
Selecting previously deselected package quota.
(Reading database ... 195153 files and directories cur-
rently installed.)
Unpacking quota (from .../quota_4.00~pre1-6_amd64.deb)
...
Processing triggers for man-db ...
Setting up quota (4.00~pre1-6) ...
```

SAIBA MAIS

Se você estiver interessado em saber mais informações sobre cotas (quotas) em sistemas operacionais Linux, acesse o *site*: <http://tldp.org/HOWTO/Quota.html> e confira as informações disponíveis.

Após a instalação da ferramenta, basta ir até o "/etc/fstab" e inserir os parâmetros "usrquota" e "grpquota" na linha que define os discos e as partições que você deseja que o sistema de cotas atue. Desta forma, agora a partição suporta o sistema de cotas. O administrador deve usar as ferramentas de gerenciamento de cotas para: checagem (quotacheck); edição de cotas (edquota); desligar e ligar sistema de cotas (quotaon e quotaoff); verificar como está o limite de uso (quota); entre outros comandos relacionados à ferramenta.

RECAPITULANDO

Neste capítulo, você conheceu os aspectos relacionados a servidores de arquivos. Foram vistos os tipos de servidores de arquivos e o protocolo SMB/CIFS, no qual você acompanhou um exemplo de configuração de compartilhamento baseado em Windows 2008 Server e em Linux, usando o Samba em um sistema operacional Debian GNU/Linux na versão *Squeeze*.

Acompanhou, ainda, a demonstração dos sistemas de cotas que podem ser utilizados tanto no Windows quanto no Linux, usando o sistema "quota". Por fim, foi demonstrado o protocolo *Network File System* (NFS) para sistemas UNIX e um exemplo de instalação para este tipo de servidor de arquivos.

Anotações:

Servidor de Impressão

8

Neste capítulo, você conhecerá aspectos relacionados a servidores de impressão, incluindo os três tipos de arquitetura para este tipo de serviço, além de conferir um exemplo de instalação e configuração de um servidor de impressão, baseado no CUPS sob a plataforma Debian GNU/Linux na versão *Squeeze*. A compreensão deste tipo de servidor e das arquiteturas envolvidas é de extrema importância para que, desta forma, o administrador de sistemas saiba como empregar as melhores tecnologias de servidores de impressão para cada caso em sua organização.

Ao final deste capítulo, você terá subsídios para:

a) entender os aspectos relacionados a servidores de impressão;

b) conhecer as arquiteturas de servidores de impressão;

c) compreender as opções para servidores de impressão.

[1] INTEROPERABILIDADE

é a capacidade de um sistema (informatizado ou não) de se comunicar de forma transparente (ou o mais próximo disso) com outro sistema (semelhante ou não).

8.1 ARQUITETURAS

Atualmente, mesmo com todos os esforços dos ativistas ambientais em conscientizar a população para evitar a impressão de arquivos, a quantidade ainda assusta. As organizações ainda têm necessidade de imprimir arquivos para registro, trâmite de processos internos, arquivos, etc. Desta forma, é importante termos em mente que a utilização de impressoras é necessária e devemos ser capazes de gerenciá-las.

Hoje em dia, é muito comum vermos os servidores de impressão nas organizações. Estes servidores vão desde máquinas que compartilham impressoras plugadas ou não, até servidores de impressão físicos, que são embutidos nas impressoras, ou ainda, podem ser vendidos como peças complementares.

Veja um exemplo de um *hardware* de um servidor de impressão.

fotógrafo

Figura 17 - Exemplo de servidor de impressão (*hardware*)

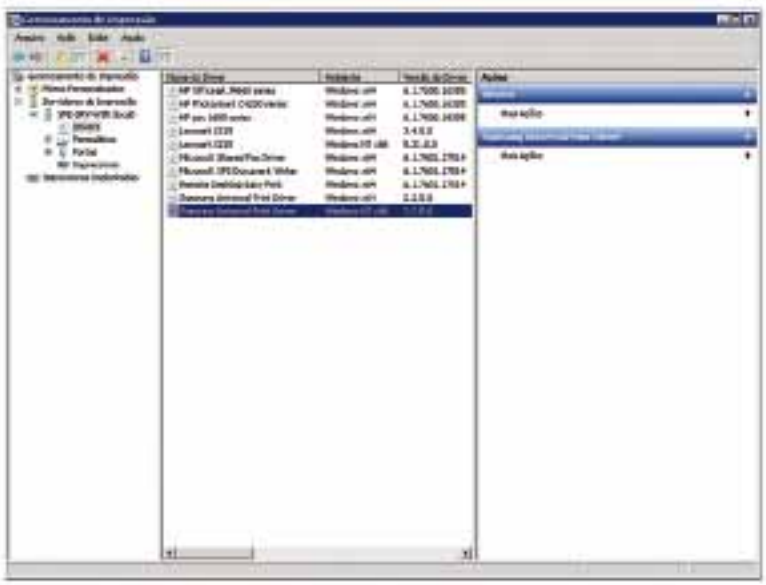
A figura mostra um servidor de impressão da empresa D-Link que tem a função de se integrar à impressora e compartilhá-la por meio da rede. Este tipo de equipamento vem com um *software* de gerência que pode ter várias características de controle e operação do *hardware*. A parte de controle pode ser feita via usuário, buscando informações de quotas de impressão, como por exemplo: o usuário João só pode imprimir 10 páginas por dia. Este tipo de *hardware* é bem popular, devido a sua simplicidade e relação custo/benefício.

Entretanto, na área de servidores de rede nós temos os Servidores de Impressão, que são importantes componentes nesta arquitetura. Estes componentes são utilizados para gerenciar impressoras e compartilhá-las. São munidos de estruturas de controle para proporcionar altos níveis de interação e facilidade de utilização por parte dos usuários. Entetanto, nem sempre foi tão fácil assim, como veremos no decorrer do capítulo.

O servidor de impressão, devido a sua amplitude de sistemas operacionais suportados e *hardwares* adequados para este fim, se divide em basicamente 3 tipos de arquiteturas: arquitetura fechada, arquitetura aberta e arquiteturas baseadas em *hardware*. Confira melhor cada uma delas.

As **arquiteturas fechadas** são conhecidas como aquelas em que as soluções de *softwares* que gerenciam o processo são de código fechado e, na grande maioria dos casos, são soluções pagas. A escolha por este tipo de solução vem ao encontro do parque computacional utilizado e, principalmente, à estrutura dos sistemas operacionais que são utilizados na organização.

Como exemplo deste tipo de arquitetura de *softwares* para construção de servidores de impressão, podemos citar a *Printer Service Tools*, para Windows 2008 Server, vista na figura a seguir.



fotógrafo

Figura 18 - Servidor de Impressão para Windows 2008 Server

As **arquiteturas abertas** para servidores de impressão são aquelas definidas como tendo livre distribuição, sem custo, e que, na maioria das vezes, têm o código fonte aberto e suas especificações disponíveis para fins de interoperabilidade[1]. Elas têm entrado em forte utilização nos últimos anos, com o crescimento na adoção dos sistemas operacionais Linux como servidores de rede. Como exemplo desta ferramenta podemos citar o CUPS, que você verá logo mais.

Há, também, as arquiteturas de servidores de impressão **baseadas em *hardware***. Estas arquiteturas são, em sua maioria, proprietárias, e têm código fechado e *softwares* para gerência do ambiente como componentes. Se popularizaram nos últimos anos, visto que os próprios fabricantes de impressora inserem em seus equipamentos essa funcionalidade. Como exemplo de empresas que criam esse tipo de solução podemos citar a HP, EPSON, D-Linl, Samsung, entre outras.

Mas, além de conhecer os tipos de arquiteturas existentes, é preciso conhecer as opções de servidores de impressão. É isso que você verá no item a seguir. Bom estudo!

8.2 OPÇÕES DE SERVIDORES IMPRESSÃO

Os sistemas Linux suportam uma grande quantidade de *software* para impressão e a maioria deles é implementada em cima do protocolo Line Printer Daemon (LDP), que é um protocolo muito flexível, porém antigo, e considerado ainda primitivo se comparado a soluções proprietárias e mais modernas para sistemas Linux. Podemos citar 3 opções de sistemas de impressão para Linux: LPD, LPRng e CUPS.

O servidor LPD original tem sido, ao longo do tempo, o padrão para sistemas Linux e, como foi assumido como padrão, muitas ferramentas para sistemas operacionais desta natureza supõem que este é o sistema padrão de impressão. Desta forma, outras versões de *softwares* de impressão para Linux têm níveis de emulação para LPD para fins de compatibilidade e bom funcionamento. Entretanto, apesar da popularidade, este sistema tem caído em desuso devido às limitações sobre integração com outros *softwares* mais modernos. O LPD, às vezes, é chamado de LPR, ou ainda, BSD LPD.

O servidor de impressão LPRng foi criado com a ideia de ser um substituito natural ao servidor LPD. O acrônimo vem de *Line Printer Remote* new generation, ou seja, um LPR de nova geração. Da mesma forma que os principais complementos ao sistema original, ele vem da área de gerência do servidor de impressão, principalmente quando estamos trabalhando em múltiplas redes.

FIQUE ALERTA

A escolha de um servidor de impressão deve levar em consideração vários aspectos técnicos sobre arquitetura computacional da empresa e ainda, deve estar alinhada com o que a empresa deseja. Desta forma, fique alerta aos requisitos necessários para este tipo de servidor!

O *Commom UNIX Printing System*, ou simplesmente CUPS, é um sistema modular de impressão para sistemas operacionais Unix-like que permite que o computador desempenhe funções de servidor de impressão. Basicamente, um computador que está executando o CUPS recebe solicitações de impressão de clientes e as processa adequadamente para as configurações estabelecidas. O primeiro CUPS disponibilizado em 1999 usava ainda o protocolo LPD como base, porém, devido a incompatibilidades, o *Internet Printing Protocol* (IPP) foi usado como padrão a partir de então. Ele suporta uma grande gama de sistemas operacionais e ainda é o sistema de impressão padrão para sistemas operacionais da família Mac OS X.

Os sistemas operacionais Windows têm como servidor de impressão nativo um componente da própria arquitetura. As versões XP e Vista vêm com um sistema de impressão local, com uma variedade de *drivers* para dispositivos de impressão de muitos fabricantes. Há ainda, a possilidade de compartilharem esta impressora em uma rede baseada nesta arquitetura.

Para as versões Server, no Windows 2003 Server foi inserido um componente chamado *Printer Service Tools*, que é uma suíte de ferramentas para gerência, instalação e compartilhamento de impressoras. Da mesma forma, é uma plataforma fechada e não tem seu código fonte disponível para estudo e melhoramentos.

Acompanhe um exemplo de caso sobre cotas em servidores de impressão. É mais um casos e relatos para você entender melhor o assunto.

CASOS E RELATOS

Cotas em servidores de impressão

O administrador da rede de uma empresa de médio porte da área de contabilidade vinha enfrentando sérios problemas com a gerência do ambiente de impressão. A cada grupo de 10 máquinas, havia um computador com uma impressora plugada sendo compartilhada. Todas as estações de trabalho da empresa usavam o Windows XP. Como o administrador entregou o controle do ambiente de impressão para as estações de trabalho, acabou perdendo a gerência das impressoras, seus estados, *drivers*, etc.

Com isso, não era mais possível precisar a porcentagem de impressões que estavam sendo realizadas para o trabalho de fato, nem quais eram os usuários que estavam imprimindo, os horários e os propósitos. Sendo assim, foi solicitado ao administrador que implantasse um sistema de controle que gerenciasse as impressões e proporcionasse informações sobre a utilização do ambiente. O administrador fez algumas pesquisas e a solução encontrada foi o CUPS, em conjunto com a PyKota, que é uma excelente ferramenta de cota de utilização e contabilização de custos. Agora, o administrador pode ter total controle sobre o sistema, podendo gerenciar as impressões e ainda extrair relatórios semanais para os gerentes acompanharem as quantidades de folhas imprimidas.

Viu o quanto as cotas podem ser práticas e úteis nos servidores de impressão? Elas podem, inclusive, ajudar a reduzir custos para a empresa. Confira, no próximo item, o exemplo de uma instalação de um servidor de impressão.

SAIBA MAIS

O tema cotas de impressão é bem interessante e útil para um administrador de rede. Para você saber mais sobre o sistema PyKota, citado no casos e relatos, consulte o *site*: <www.pytoka.com>.

8.3 EXEMPLO DE INSTALAÇÃO DE UM SERVIDOR DE IMPRESSÃO

Para essa instalação do servidor de impressão, será usado o CUPS como exemplo, devido a sua grande popularidade e por atingir um maior número de sistemas operacionais suportados.

O exemplo será executado tendo como padrão um sistema operacinal Debian GNU/Linux na versão *Squeeze*. No código a seguir é possível ver a instalação do CUPS usando o sistema gerenciador de pacotes APT.

```
root@server:/home/douglas# apt-get install cups
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
Os pacotes extra a seguir serão instalados:
  cups-client cups-common cups-driver-gutenprint cups-ppdc dbus defoma fancontrol fontconfig-config foomatic-db foomatic-db-engine foomatic-filters foomatic-filters-ppds ghostscript ghostscript-cups gsfonts hpijs libavahi-client3
   libavahi-common-data libavahi-common3 libcups2 libcupscgi1 libcupsdriver1 libcupsimage2 libcupsmime1 libcupsppdc1 libdbus-1-3 libfont-freetype-perl libfontconfig1 libfontenc1 libgs8 libgutenprint2 libhpmud0 libijs-0.35 libjasper1
   libjbig2dec0 libjpeg62 liblcms1 libopenjpeg2 libpaper-utils libpaper1 libperl5.10 libpng12-0 libpoppler5 libsensors4 libslp1 libsnmp-base libsnmp15 libtiff4 libxfont1 lm-sensors min12xxw pnm2ppa poppler-utils ttf-dejavu-core
   ttf-freefont x-ttcidfont-conf x11-common xfonts-encodings xfonts-utils
Pacotes sugeridos:
   cups-bsd hplip xpdf-korean xpdf-japanese xpdf-chinese-traditional xpdf-chinese-simplified cups-pdf smbclient kdeprint gtklp cups-pt xpp gutenprint-doc gutenprint-locales dbus-x11 defoma-doc psfontmgr dfontmgr hplip-cups foo2zjs splix
   m2300w pxljr openprinting-ppds openprinting-ppds-extra cjet foomatic-db-gutenprint hpijs-ppds hplip-doc libjasper-runtime liblcms-utils slpd openslp-doc snmp-mibs-downloader sensord read-edid i2c-tools magicfilter apsfilter
Os NOVOS pacotes a seguir serão instalados:
   cups cups-client cups-common cups-driver-gutenprint cups-ppdc dbus defoma fancontrol fontconfig-config foomatic-db foomatic-db-engine foomatic-filters foomatic-filters-ppds ghostscript ghostscript-cups gsfonts hpijs libavahi-client3
```

```
    libavahi-common-data libavahi-common3 libcups2 lib-
cupscgi1 libcupsdriver1 libcupsimage2 libcupsmime1 lib-
cupsppdc1 libdbus-1-3 libfont-freetype-perl libfontconfig1
libfontenc1 libgs8 libgutenprint2 libhpmud0 libijs-0.35
libjasper1
     libjbig2dec0 libjpeg62 liblcms1 libopenjpeg2 lib-
paper-utils libpaper1 libperl5.10 libpng12-0 libpoppler5
libsensors4 libslp1 libsnmp-base libsnmp15 libtiff4 libx-
font1 lm-sensors min12xxw pnm2ppa poppler-utils ttf-deja-
vu-core
    ttf-freefont x-ttcidfont-conf x11-common xfonts-en-
codings xfonts-utils
  0 pacotes atualizados, 60 pacotes novos instalados, 0 a
serem removidos e 2 não atualizados.
  É preciso baixar 30,8 MB/30,8 MB de arquivos.
  Depois desta operação, 101 MB adicionais de espaço em
disco serão usados.
  Você quer continuar? (S/N)
```

Após a instalação das ferramentas, os arquivos de configuração estarão disponíveis no diretório "/etc/cups", como você verá no código a seguir.

VOCÊ SABIA?

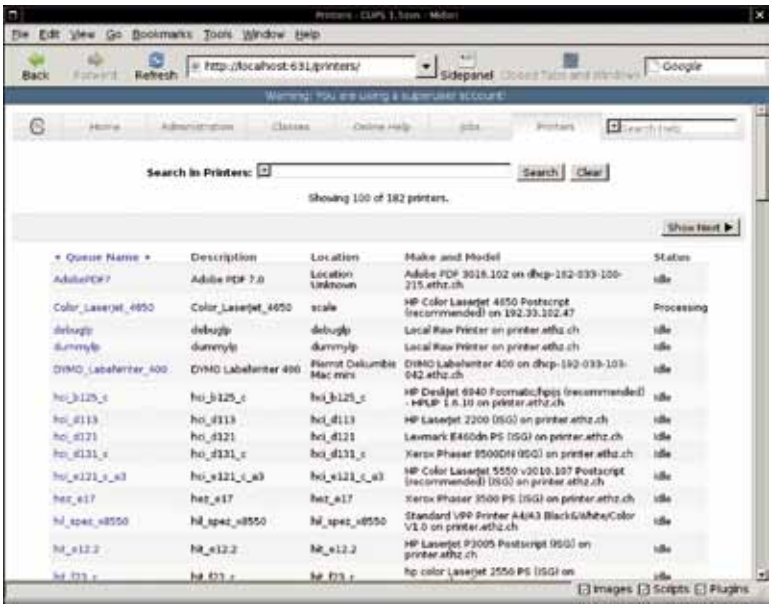
Utilizando o CUPS, você pode compartilhar impressoras entre múltiplos sistemas operacionais. Mais informações sobre esta importante ferramenta podem ser encontradas em: <www.cups.org>.

Neste arquivo, você poderá configurar todas as características que espera da ferramenta, inserir *drivers* de dispositivos, tipos de impressão, configurar agentes de monitoramento e, ainda, a parte de segurança, que é baseada na *Secure Socket Layer* (SSL).

```
root@server:/etc# ls -l cups/
total 32
-rw-r--r-- 1 root root 4058 Nov 12  2010 cupsd.conf
-rw-r--r-- 1 root root 4178 Nov 12  2010 cupsd.conf.
default
drwxr-xr-x 2 root lp   4096 Nov 12  2010 ppd
-rw-r--r-- 1 root root  240 Set 25 16:06 raw.convs
-rw-r--r-- 1 root root  211 Set 25 16:06 raw.types
-rw-r--r-- 1 root root  186 Nov 12  2010 snmp.conf
drwx------ 2 root lp   4096 Set 25 16:06 ssl
```

Apesar de podermos gerenciar todas as funções do CUPS por linha de comando, ele vem com uma excelente interface de configuração, que pode ser acessada via *browser* e usada para incluir ou remover impressoras, gerenciar filas de impressão, entre outras funções. O acesso pode ser feito via *browser* convencional, usando o endereço IP, seguido da porta padrão do CUPS (631 TCP).

Veja um exemplo da tela de gerência via Web do CUPS.



fotógrafo

Figura 19 - Interface Web do CUPS

RECAPITULANDO

Você conheceu aspectos relacionados aos servidores de impressão. Viu as arquiteturas abertas e fechadas, e aquelas que têm como base o *hardware* e que podem compor este tipo de ambiente, mas que ainda usam *softwares* para gerência. Acompanhou exemplos de servidores de impressão, tais como o LPD, LPRng, CUPS e o Printer Service Tools. Por fim, conferiu a instalação de um exemplo do CUPS, tendo como base um Debian GNU/Linux.

Lembre-se de que o aprendizado não pode parar. Dedique-se ao máximo, pesquise e busque novidades e atualizações, pois, em se tratando de tecnologias, as mudanças são constantes.

Anotações:

Servidor DHCP

9

Neste capítulo, você verá detalhes do serviço de configuração automática de um servidor DHCP, passando por sua história, modo de operação e os tipos de alocação que este tipo de servidor trabalha. Conhecerá um pouco das ferramentas que podemos implementar o protocolo e, ainda, acompanhará um exemplo de instalação usando o servidor DHCP da ISC, que é o mesmo mantenedor do Bind9 (DNS).

Ao final deste capítulo, você terá subsídios para:

a) conhecer a evolução e a história dos servidores DHCP;

b) entender o modo de operação dos servidores DHCP;

c) conhecer as ferramentas para servidores DHCP.

9.1 HISTÓRIA

Antes de conhecer mais profundamente o servidor DHCP, que tal entender como tudo começou? A tarefa de configuração manual do TCP/IP em máquinas clientes é um fardo pesado para usuários iniciantes. Desta forma, os desenvolvedores trabalham para diminuir este peso, desenvolvendo algoritmos que configurem automaticamente as máquinas dos usuários. Isto simplifica a configuração das estações e o trabalho dos administradores de sistema, que não precisam reconfigurar de máquina em máquina, todas as estações a cada nova alteração que for realizada.

Para desempenhar esta função, foi criado o Protocolo de Configuração Dinâmica de Máquinas ou *Dynamic Host Configuration Protocol* (DHCP), que é um protocolo de configuração de rede para máquinas que trabalham com redes baseadas no IP. Os computadores que estão interligados a redes IP devem ser configurados antes que eles possam se comunicar com outras máquinas da rede. As informações mais básicas necessárias são: um endereço IP, uma rota padrão e um prefixo de roteamento.

Além de endereços IP, os servidores DHCP também podem fornecer outras informações mais avançadas de configuração, como por exemplo: servidores de nomes (DNS) primários e secundários, nomes de máquinas, servidores de inicialização (*boot*), controle de endereçamentos baseados em endereços MAC, entre outros serviços para configuração dinâmica das máquinas dos clientes da rede.

Um exemplo de arquitetura DHCP pode ser visto na figura a seguir.

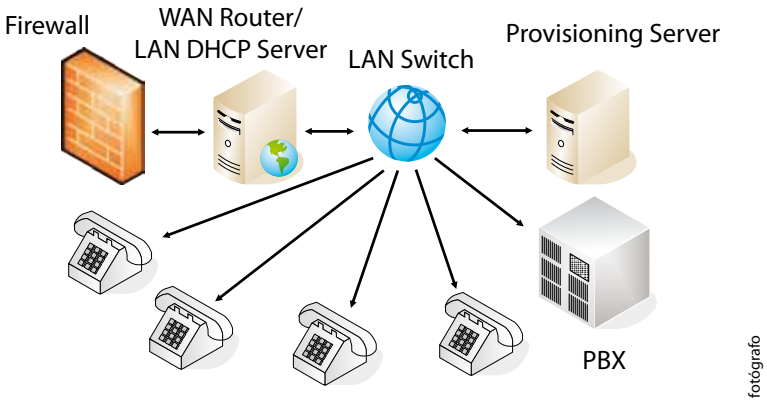


Figura 20 - Exemplo de arquitetura DHCP

O serviço de configuração automática de hosts pode ser usado tanto para IPv4 como para IPv6. Um fato interessante quanto a estas duas abordagens é que, embora ambas desempenhem o mesmo papel em uma rede de computadores, os detalhes internos são muito diferentes e podem ser considerados como protocolos praticamente distintos.

Em outubro de 1993, a RFC 1531 foi lançada e nela foi definido o protocolo DHCP, que nasceu como uma extensão para outro protocolo, chamado BOOTP ou Protocolo de Bootstrap. A necessidade para a criação deste novo protocolo veio advindo de problemas relacionados à manutenção dos servidores BOOTP que necessitavam, entre outras coisas, de intervenção manual a cada nova máquina que fosse ingressar na rede.

Em 1997, a RFC 2131 foi lançada a fim de esclarecer algumas nuances do protocolo e, até hoje, esta RCF continua ser o padrão para redes IPv4. Para redes IPv6, o protocolo DHCP está documentado na RFC 3315 e algumas melhorias para esta RFC foram propostas na RFC 3633 e 3736. Sendo ambas modificações baseadas em extensões de melhorias para o padrão.

Agora que você conheceu um pouco da história, vamos conhecer o servidor. Para começar, conheça o modo de operação. Esse é o assunto do item seguinte.

9.2 MODO DE OPERAÇÃO

Os servidores DHCP utilizam, por padrão, duas portas: 67 (UDP) e 68 (UDP). A primeira porta é usada para comunicação com o servidor e a segunda, para comunicação com o cliente. As DHCP dividem-se em: 1) descoberta IP, 2) oferta IP, 3) solicitação IP e 4) reconhecimento IP. Toda a comunicação e operações relacionadas funcionam via UDP.

Quando um cliente configurado para buscar seu endereço IP via protocolo DHCP se conecta à rede, o cliente envia uma solicitação buscando informações básicas do servidor DHCP. Ao receber o pedido válido de um cliente, o servidor atribui ao cliente que solicitou: um endereço IP, um lease (tempo da alocação de IP) e outros parâmetros de configuração IP, tais como máscara, *gateway*, endereço de *broadcast*, entre outros.

Os servidores DHCP podem ter até três métodos de alocação de endereços IP, que dependerão da implementação do protocolo mas, por padrão, podem ser:

a) **alocação dinâmica**: quando o administrador atribui um intervalo (*range*) de endereços IP para o servidor DHCP no momento em que cada cliente da rede já está configurado para solicitar informações ao servidor DHCP. Este processo de alocação usa o conceito de *lease* para possíveis realocações e para termos um maior controle sobre a rede;

b) **alocação automática**: é quando o servidor DHCP determina um endereço IP para um cliente que foi atribuído pelo administrador. Funciona como na alocação dinâmica mas, nesta forma de alocação, o servidor DHCP mantém uma tabela de atribuições de endereço de modo que ele possa atribuir a um cliente o mesmo endereço que ele atribuiu na outra vez;

c) **alocação estática**: quando o servidor DHCP atribui um endereço IP com base em uma tabela de endereços MAC. Desta forma, sempre que um cliente com o endereço MAC já registrado solicitar um IP, ele sempre terá o mesmo endereço que foi determinado pelo administrador de rede. É muito utilizado em caso de servidores de rede que não podem ficar trocando de endereço IP constantemente.

FIQUE ALERTA

Preste muita atenção na infraestrutura de rede de sua organização, pois, um servidor DHCP mal configurado pode representar uma falha de segurança grave para a rede.

9.3 FERRAMENTAS PARA SERVIDORES DHCP

Como nos casos de outras tecnologias, podemos encontrar vários *softwares* que implementam o protocolo para disponibilização do serviço. Estes *softwares* variam em questões de suporte a plataformas, características de segurança, funcionalidades, entre outros quesitos.

No quadro a seguir, é possível visualizar alguns exemplos de servidores DHCP.

FERRAMENTA	PLATAFORMAS
ISC DHCP Server	Linux, Unix, Mac OS X, BSD
Udhcpd	Linux, Mac OS X, BSD
DHCP Server	Windows
DHCP Turbo	Windows

Quadro 9 - Ferramentas para Servidores DHCP

Chegou a hora de conferir mais um Casos e relatos, para entender melhor o assunto. Ajeite-se confortavelmente e boa leitura!

CASOS E RELATOS

O estudante desatento

Certa vez, em uma cliníca na área de imagens médicas, o administrador se deparou com um problema muito interessante. Em sua estrutura, havia um sevidor DHCP que distribuía IPs da faixa 192.168.5.0/24, ou seja, todos os endereços dos clientes deveriam iniciar com 192.168.5. Certa manhã ele foi chamado para resolver um problema em uma estação cliente na qual o usuário não conseguia navegar. A primeira coisa que o administrador checou foi a parte de configuração de endereçamento da máquina e constatou que ela usava DHCP, porém, o DHCP atibuiu a essa máquina o IP 10.1.1.51, juntamente com todas as configurações de *broadcast*, *gateway*, entre outros. Essa cituação curiosa deixou o administrador intrigado, pois o IP estava fora da faixa correta. Pouco tempo depois, o mesmo problema repetiu-se em outra máquina e, ao final daquele dia, metade dos clientes estavam reclamando do mesmo problema (não era possível navegar).

Ao tentar renovar o IP dos clientes, o administrador percebeu que era pego o IP errado. Ele decidiu, então, usar uma ferramenta de *sniffing* de rede para analisar o fluxo e UDPO da rede a cada nova solicitação e percebeu que, possivelmente, havia outra máquina como servidor DHCP, e isso estava causando todos os problemas. Rastreando o IP do possível servidor DHCP não autorizado ele chegou ao nome "João". João era o rapaz que cuidava da ressonância magnética e estava fazendo um curso de redes de computadores. Ao ser questionado, João disse que estava estudando como montar um servidor DHCP e seu *notebook* estava plugado na rede. Como João estava fazendo tudo certinho, seus testes estavam interferindo na rede da clínica. O administrador foi então até a sala de João, desligou o serviço, e tudo voltou a funcionar perfeitamente na rede.

Pois é... devemos tomar sempre muito cuidado com essa tecnologia para não prejudicar o andamento da rede. Para evitar problemas como o causado pelo João, do casos e relatos, é importante trabalhar com tecnologias de Redes Virtuais (VLANs) nos *switches*. Assim, você fica preso a um determinado enlace e não corre o risco de sair distribuindo IPs pela rede.

E, já que estamos falando em instalação de servidor DHCP, confira no item a seguir, um exemplo básico de instalação.

9.4 INSTALAÇÃO DE UM SERVIDOR DHCP

Para este exemplo de instalação de um servidor DHCP, usaremos o ISC DHCP *Server da Internet Systems Consortium* (ISC) como base, que é a mesma mantenedora e distribuidora do DNS *Server* e *Bind*, que você já viu nos capítulos anteriores. A instalação será realizada tendo um servidor Debian Linux/GNU versão *Squeeze* como base. Confira os comandos a seguir. Nessa instalação foi utilizado o gerenciador de pacotes "apt-get".

```
  root@server:/# apt-get install dhcp3-server
  Lendo listas de pacotes... Pronto
  Construindo árvore de dependências
  Lendo informação de estado... Pronto
  Os pacotes extra a seguir serão instalados:
    isc-dhcp-server
  Pacotes sugeridos:
    isc-dhcp-server-ldap
  Os NOVOS pacotes a seguir serão instalados:
    dhcp3-server isc-dhcp-server
  0 pacotes atualizados, 2 pacotes novos instalados, 0 a
serem removidos e 2 não atualizados.
  É preciso baixar 436 kB de arquivos.
  Depois desta operação, 897 kB adicionais de espaço em
disco serão usados.
  Você quer continuar [S/n]? S
  Obter:1  http://security.debian.org/  squeeze/updates/
main isc-dhcp-server amd64 4.1.1-P1-15+squeeze3 [410 kB]
  Obter:2  http://security.debian.org/  squeeze/updates/
main dhcp3-server all 4.1.1-P1-15+squeeze3 [25,9 kB]
  Baixados 436 kB em 1s (407 kB/s)
  Pré-configurando pacotes ...
  Selecionando pacote previamente não selecionado isc-
dhcp-server.
  (Lendo banco de dados ... 27514 ficheiros e directórios
actualmente instalados.)
  Desempacotando  isc-dhcp-server  (de  .../isc-dhcp-
server_4.1.1-P1-15+squeeze3_amd64.deb) ...
  Selecionando pacote previamente não selecionado dhcp3-
server.
```

```
  Desempacotando     dhcp3-server     (de     .../dhcp3-
server_4.1.1-P1-15+squeeze3_all.deb) ...
  Processando gatilhos para man-db ...
  Configurando isc-dhcp-server (4.1.1-P1-15+squeeze3) ...
  Generating /etc/default/isc-dhcp-server...
  Starting ISC DHCP server: dhcpd/et     icheck   syslog
for diagnostics. ...
  invoke-rc.d: initscript isc-dhcp-server, action "start"
failed.
  Configurando dhcp3-server (4.1.1)
```

Após a instalação do servidor DHCP, os arquivos de configuração estarão disponíveis para customização no diretório "/etc/dhcp". Dentro deste diretório, há o arquivo principal de instalação do servidor DHCP, chamado "dhcp.conf".

A seguir, você pode ver um exemplo de "dhcp.conf" funcional.

```
  # dhcpd.conf
  #
  # Configuration file for ISC dhcpd (see 'man dhcpd.conf')
  #

  authoritative;
  Default-lease-time -1;
  max-lease-time -1;
  option subnet-mask 255.255.255.0;
  option broadcast-address 192.168.1.255;
  option routers 192.168.1.254;
  option domain-name-servers 192.168.1.1, 192.168.1.2;
  option domain-name "exemplo.com";

  #Logging;
  log-facility local7;

  subnet 192.168.1.0 netmask 255.255.255.0 {
     default-lease-time 86400;
     max-lease-time 86400;
```

```
    range 192.168.1.60 192.168.1.160;
}

#
# Maquinas registradas
#

host host1 {
    hardware ethernet 00:04:23:b4:69:42;
    fixed-address 192.168.1.7;
}

host host2 {
    hardware ethernet 00:04:23:b4:69:42;
    fixed-address 192.168.1.8;
}

host host3 {
    hardware ethernet 00:04:23:b4:69:42;
    fixed-address 192.168.1.9;
}

host host4 {
    hardware ethernet 00:04:23:b4:69:42;
    fixed-address 192.168.1.10;
}
```

VOCÊ SABIA?

Você sabia que, mesmo em uma rede que está sob gerência de um servidor DHCP, o servidor pode "fixar" IPs em determinados *hosts* baseados em seus endereços físicos (MAC)?

No arquivo “dhcp.conf” é possível fazer várias configurações e ajustes para suportar sua infraestrutura. No exemplo dado, você pode visualizar que o servidor DHCP está provendo endereços IP dinamicamente da faixa entre os IPs 192.168.1.60 até o 192.168.1.160. Ainda, pode-se visualizar que existem 3 casos de servidores que estão com IPs “fixos” relacionados por MAC, o que significa que toda vez que o servidor DHCP receber uma solicitação de endereços destes MACs, os IPs relacionados serão entregues.

SAIBA MAIS

Na página oficial da documentação da ferramenta ISC DHCP *Server* é possível ver todos os parâmetros existentes do arquivo. O endereço é: <http://www.isc.org>. Acesse!

No quadro a seguir, é possível visualizar exemplos dos parâmetros que este arquivo suporta.

PARÂMETRO	DESCRIÇÃO
default-lease-time	Este parâmetro determina o tempo limite da concessão do endereço IP para os hosts.
max-lease-time	Parâmetro que determina que, se o cliente solicitar um tempo maior, o tempo máximo permitido será o fixado neste parâmetro.
options subnet-mask	Define a máscara de sub-rede a ser fornecida aos clientes, juntamente com o seu IP.
options broadcast-address	É o endereço de envio para requisições de broadcast.
options routers	O cliente, além do endereço IP, receberá também a informação do endereço IP do equipamento que será o seu gateway da rede.
options domain-name-servers	Esta opção lista os servidores de nomes (DNS) a serem utilizados para resolução de nomes.
options domain-name	As máquinas que solicitarem, devem pertencer a um domínio, desta forma, o DHCP Server fornecerá o nome de domínio da máquina.
subnet	Esta opção determina a sub-rede que o servidor DHCP estará trabalhando.
range	Este parâmetro determina o intervalo (range) de endereços que o servidor DHCP fornecerá aos seus clientes.
host	Parâmetro geralmente utilizado para fixar endereços IPs em máquinas, vinculados ao MAC Address.

Quadro 10 - Parâmetros e descrições do “dhcp.conf”

RECAPITULANDO

Neste capítulo, você estudou aspectos e características relacionadas a servidores que implementam o Protocolo de Configuração Dinâmica de Máquinas ou, simplesmente, DHCP. Conheceu a sua história, o modelo de operação, os tipos de alocação, viu alguns exemplos sobre ferramentas que implementam este protocolo e, por fim, conferiu um exemplo de instalação deste tipo de servidor, no qual usamos o ISC DHCP *Server*. Ainda tem muita coisa para você aprender nessa área, portanto, continue atento!

Anotações:

Servidor de Conexão Remota

10

Em termos de serviços, com o tempo há necessidade de termos servidores de conexão remota para acessar máquinas remotamente. Os motivos para o surgimento deste tipo de tecnologia foram vários, mas a maioria deles estava vinculado a gerência de máquinas. Desde o início dos sistemas computacionais em rede, já haviam soluções de acesso remoto para os mainframes, e estas tecnologias foram evoluindo de acordo com as demandas dos clientes, no sentido de manter e gerenciar os servidores de redes.

Neste capítulo, você conhecerá aspectos técnicos dos protocolos que proporcionam acesso remoto a servidores e máquinas que compõem uma rede. Ao final deste capítulo, você terá subsídios para:

a) conhecer os aspectos do servidores de conexão remota;

b) entender o protocolo RDP;

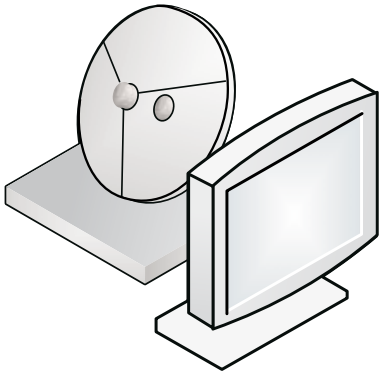
c) conhecer o VNC;

d) compreender as funcionalidades do SSH;

e) entender os servidores Telnet.

10.1 RDP

O Protocolo Remoto de *Desktop* ou *Remote Desktop Protocol* (RDP) é um protocolo proprietário que foi desenvolvido pela Microsoft e que fornece a um usuário a possibilidade de uma interface gráfica remota para outro computador. O protocolo é uma extensão do ITU-T T.128 que é um protocolo de compartilhamento de aplicativos. Para este protocolo, existem clientes à maioria das versões do Microsoft Windows (incluindo Windows Mobile), Linux , Unix , Mac OS X , Android e outros modernos sistemas operacionais. Por padrão, o servidor escuta em TCP porta 3389.



fotógrafo

Figura 21 - Figura ilustrativa do RDP

Atualmente, a Microsoft refere-se a seu *software* oficial RDP como *Remote Desktop Services* ou Serviço de Desktop Remoto. Para plataforma Windows, o *software* cliente oficial é atualmente referida como o *Remote Desktop Connection*. Este *software* é amplamente difundido, tanto para manutenção e gerência de servidores, mas bem como tarefas administrativas em máquinas clientes de uma rede de computadores.

Na figura a seguir, você pode ver a tela de acesso ao RDP em um cliente Windows XP.

fotógrafo

Figura 22 - Cliente de acesso ao RDP

Para cliente que não são da plataforma Windows existem muitas implementações de clientes e servidores RDP. Para clientes baseados em Linux, há o "rdesktop", que é muito utilizado para acessar máquinas Windows a partir de clientes Linux. Exitem também variantes como o KRDC e o "tsclient" que são construídos em cima do "rdesktop", já que ele tem seu código-aberto e permite este tipo de customização e alteração.

Para acessar um cliente remoto, usando o "rdesktop", pode usar o comando a seguir:

```
$ rdesktop 192.168.1.250
```

No ano de 2009 o "rdesktop" teve uma modificação em seu sistema de desenvolvimento, que originou o FreeRDP. O FreeRDP é um projeto que tem o objetivo de modularizar o código da ferramenta, dando foco em algumas questões técnicas, bem como a implementação de novas funcionalidades para ferramenta. Há ainda uma excelente compatibilidade que é o fato de podermos, a partir de um host Windows, usando o Windows Desktop Connection, conectarmos a um servidor rodando FreeRDP.

Você acabou de conhecer um protocolo que foi desenvolvido para acessar máquinas remotamente, porém, não existe somente esse protocolo. Conheça mais um, no item a seguir.

10.2 VNC

O *Virtual Network Computing* (VNC) foi originalmente desenvolvido pelo Laboratório de Pesquisa Olivetti, na cidade de Cambridge, no Reino Unido. Em termos básicos o VNC é um sistema de compartilhamento de desktop que usa o protocolo RFB (*Framebuffer Remoto*) para controlar remotamente outro computador. Uma de suas principais características é que ele transmite os eventos do *mouse* e do teclado de um computador para outro. Desta forma, é possível controlar totalmente um computador remotamente.

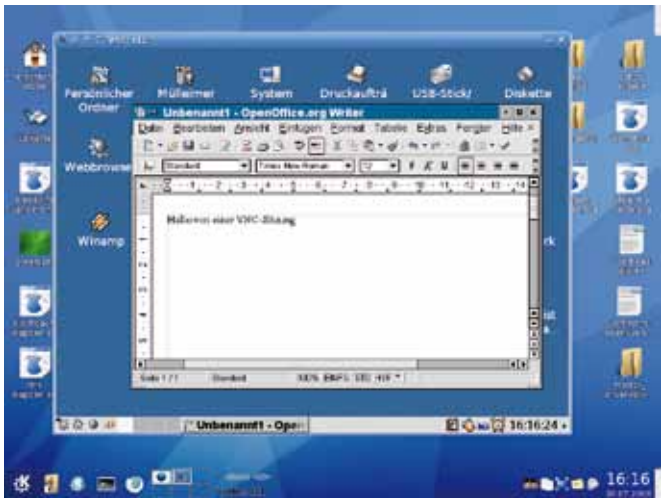
O VNC não depende de plataforma computacional, por isso é possível um cliente VNC, a partir de um sistema operacional Linux, por exemplo, conectar-se a um servidor VNC em um sistema operacional Windows. Existem muitos clientes e servidores VNC para várias plataformas, que vão desde Windows, passando Linux, UNIX, Mac OS X, BSB e afins. O VNC tem muitas variantes que oferecem funcionalidades que não estão adequadas aos padrões propostos, mas que são interoperáveis entre si. O VNC e o RFB são marcas registradas da RealVNC Ltda.

fotógrafo

Figura 23 - Logo do *Virtual Network Computing*

O modo de operação de um serviço de acesso remoto a servidores e clientes passa por um pedido de conexão a um determinado *host*, na porta 5900 (TCP), que permitirá, de acordo com as métricas de segurança que foram definidas, se permite ou não a conexão ao servidor. O servidor VNC, permitindo a conexão, enviará a tela completa e o controle compartilhado do ambiente para o cliente que solicitou a conexão.

Na figura a seguir, é possível visualizar um cliente em um sistema operacional Linux, conectando-se a um servidor VNC.



fotógrafo

Figura 24 - Exemplo de Conexão Via VNC

Como servidor VNC, há uma alternativa muito utilizada para sistemas operacionais Linux chamada vnc4server. O modo de operação não muda, já que estamos falando de mais uma implementação baseada no protocolo RFB.

Observe, no código a seguir, a sua instalação.

```
root@server:/# apt-get install vnc4server
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
Os pacotes extra a seguir serão instalados:
   cpp cpp-4.4 fontconfig-config libdrm-intel1 libdrm-
radeon1 libdrm2 libfontconfig1 libfontenc1 libfs6 libgl1-
mesa-dri
```

```
    libgl1-mesa-glx libgmp3c2 libice6 libmpfr4 libpng12-0
libsm6 libutempter0 libxaw7 libxcb-atom1 libxcursor1 libx-
damage1
      libxfixes3  libxfont1  libxft2  libxi6  libxinerama1
libxkbfile1 libxmu6 libxpm4 libxrandr2 libxrender1 libxt6
libxtst6 libxv1
     libxxf86dga1  libxxf86vm1  ttf-dejavu-core  x11-apps
x11-common x11-session-utils x11-utils x11-xfs-utils x11-
xkb-utils
    x11-xserver-utils xbase-clients xbitmaps xfonts-base
xfonts-encodings xfonts-utils xinit xterm
  Pacotes sugeridos:
    cpp-doc gcc-4.4-locales libglide3 vnc-java mesa-utils
nickle cairo-5c xorg-docs-core xfs xserver xfonts-cyril-
lic
  Os NOVOS pacotes a seguir serão instalados:
     cpp  cpp-4.4  fontconfig-config  libdrm-intel1  libdrm-
radeon1 libdrm2 libfontconfig1 libfontenc1 libfs6 libgl1-
mesa-dri
    libgl1-mesa-glx libgmp3c2 libice6 libmpfr4 libpng12-0
libsm6 libutempter0 libxaw7 libxcb-atom1 libxcursor1 libx-
damage1
      libxfixes3  libxfont1  libxft2  libxi6  libxinerama1
libxkbfile1 libxmu6 libxpm4 libxrandr2 libxrender1 libxt6
libxtst6 libxv1
     libxxf86dga1 libxxf86vm1 ttf-dejavu-core vnc4server
x11-apps x11-common x11-session-utils x11-utils x11-xfs-
utils
     x11-xkb-utils x11-xserver-utils xbase-clients xbit-
maps xfonts-base xfonts-encodings xfonts-utils xinit xterm
  0 pacotes atualizados, 52 pacotes novos instalados, 0 a
serem removidos e 2 não atualizados.
  É preciso baixar 35,5 MB de arquivos.
  Depois desta operação, 80,8 MB adicionais de espaço em
disco serão usados.
```

SAIBA MAIS Para saber mais detalhes sobre o VNC, acesse a página oficial da RealVNC: <http://www.realvnc.com>.

Há um protocolo que proporciona uma comunicação segura de dados. É o *Secure Shell*, conhecido como SSH. Saiba mais sobre esse protocolo no item a seguir.

10.3 SSH

O *Secure Shell* ou *Shell* Seguro (SSH) é um protocolo de rede da pilha TCP/IP que proporciona comunicação segura de dados, provê ainda *shell* remoto seguro ou a execução de comandos e outros serviços de redes entre dois ou mais computadores, por meio de um canal seguro de comunicação. A aplicação mais conhecida do protocolo é de acesso a contas *shell* em sistemas operacionais do tipo Unix, tais como Linux, AIX, HP-UX entre outros.

A necessidade de um protocolo desta natureza surgiu devido à popularidade dos protocolos *telnet*, *rlogin* e *rexec*, e à fraca segurança que estes protocolos têm, o que culminou no surgimento do SSH. O objetivo da camada de criptografia usada pelo servidor SSH se destina a fornecer confidencialidade e integridade de dados por meio de uma rede não segura, como a Internet.

Sobre o modo de operação do SSH, ele utiliza criptografia de chave pública para fornecer autenticação de computadores remotos e permitir assim que o computador remoto possa autenticar usuários, se este for o caso. Desta forma, o SSH apenas checa se a mesma pessoa ou computador, que oferece a chave pública também possui a chave privada correspondente.

O serviço SSH é normalmente usando para acessar em máquinas remotas e executar comandos no sistema operacional, mas também suporta, entre outras coisas, tunelamento (construir um canal de comunicação seguro entre duas máquinas), encaminhamento de portas e conexões, baseadas no protocolo X11 (possibilidade de podermos visualizar o ambiente gráfico do servidor). Ainda há o comando para transporte seguro de dados SCP (*Secure Copy*).

A porta padrão de um servidor SSH é a 22 (TCP), porém, é muito comum os administradores trocarem esta porta por razões de segurança, já que muitos *crackers* fazem tentativas de acesso baseadas em força bruta para conseguir acesso ao sistema. Estes ataques geralmente são efetuados baseados em dicionários, o que nos remete à não utilização de palavras que constam no dicionário em nossas senhas.

VOCÊ SABIA?

Você sabia que atualmente os distribuidores como Cisco, 3com entre outros estão trocando todos os acessos a seus equipamentos e passando a utilizar o protocolo SSH?

Para servidores Linux, a versão mais utilizada para implantação do serviço SSH é o *OpenSSH Server*. A seguir, você pode ver um exemplo de instalação de um servidor SSH em um sistema operacional *Debian Linux/GNU* versão *Squeeze*.

```
root@server:/# apt-get install openssh-server
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
Pacotes sugeridos:
  ssh-askpass rssh molly-guard ufw
Os NOVOS pacotes a seguir serão instalados:
  openssh-server
0 pacotes atualizados, 1 pacotes novos instalados, 0 a
serem removidos e 2 não atualizados.
É preciso baixar 318 kB de arquivos.
Depois desta operação, 823 kB adicionais de espaço em
disco serão usados.
Obter:1  http://debian.pop-sc.rnp.br/debian/  squeeze/
main openssh-server amd64 1:5.5p1-6 [318 kB]
Baixados 318 kB em 0s (2080 kB/s)
Pré-configurando pacotes ...
Selecionando pacote previamente não selecionado openssh-
server.
(Lendo banco de dados ... 27553 ficheiros e directórios
actualmente instalados.)
Desempacotando  openssh-server  (de  .../openssh-
server_1%3a5.5p1-6_amd64.deb) ...
Processando gatilhos para man-db ...
Configurando openssh-server (1:5.5p1-6) ...
Creating SSH2 RSA key; this may take some time ...
Creating SSH2 DSA key; this may take some time ...
Restarting OpenBSD Secure Shell server: sshd.
```

No próximo quadro, você pode ver uma tentativa de conexão usando o SSH, onde o usuário Douglas está tentando acessar, via protocolo SSH, o servidor com o endereço 192.168.1.32. Primeiro, o servidor irá checar se o cliente tem a chave de pública de criptografia e, após esta negociação, poderá perguntar a senha do usuário no sistema.

Observe a seguir.

```
root@server:/# ssh douglas@192.168.1.32
The authenticity of host 192.168.1.32 (192.168.1.32)' can't be established.
RSA key fingerprint is 0e:9a:d1:99:18:64:b5:1a:c4:c4:ed:07:08:a4:0a:b3.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.32' (RSA) to the list of known hosts.
douglas@192.168.1.32's password:
Linux server 2.6.32-5-amd64 #1 SMP Fri Sep 9 20:23:16 UTC 2011 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
Last login: Wed Sep 14 12:39:24 2011 from 192.168.1.12
douglas@server:~$
```

Acompanhe, no Casos e relatos a seguir, um exemplo prático sobre as vantagens dos servidores de conexão remota.

CASOS E RELATOS

Economia de tempo e esforço

O setor de manutenção de computadores de um grande hospital da rede pública estava enfrentando um problema na área de suporte aos usuários finais. Como o hospital era muito grande, contava com sete andares, toda vez que um usuário ligava, o técnico precisava se deslocar até a estação do cliente. Em 75% dos casos, o problema relatado era muito simples de resolver, a maioria das vezes era operação indevida por parte do usuário. Para tentar resolver esse problema em conjunto, a equipe de administração de redes foi convocada e, após uma longa conversa, ficou definido que a partir daquele momento, todas as máquinas que passassem por revisão no setor de manutenção teriam um esquema VNC instalados. Os administradores iniciaram um processo de mapeamento de cada uma das mais de 700 máquinas do hospital, identificando-as por nome. Agora, toda vez que um cliente liga solicitando apoio, os técnicos podem acessar a estação de trabalho do setor em que estão efetuando o reparo remotamente.

Então, viu só os benefícios dos servidores de conexão remota? Eles facilitam muito o trabalho e ajudam a reduzir custos. Mas, ainda não acabou. Falta você conhecer o *Telnet*. Vamos em frente?

10.4 *TELNET*

O *Telnet* foi um dos primeiros padrões da Internet. Foi lançado, em 1969, na RFC 15, e posteriormente, estendido para a RFC 854. Ele é um protocolo de rede utilizado em redes que fornece um ambiente interativo, via texto (linha de comando) de comunicação, usando um terminal virtual de conexão. Por padrão, ele funciona sobre o TCP na porta 23.

No passado, a maioria dos equipamentos de rede e sistema operacionais, baseados em TCP/IP vinham com o serviço *Telnet* incluído, mas, por razões de segurança, o *Telnet* caiu em desuso, sendo substituído por protocolos mais seguros, como o SSH, visto anteriormente.

O modo de operação do serviço *Telnet* baseado em no paradigma cliente-servidor onde um cliente solicita, por meio de um cliente *telnet*, uma conexão na porta 23 usando o protocolo *Telnet* para acesso a um *host* remoto.

Veja a seguir, um exemplo de instalação de um servidor *Telnet* em um sistema operacional *Debian Linux/GNU* versão *Squeeze*.

```
root@server:/# apt-get install telnetd
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
Os pacotes extra a seguir serão instalados:
  libfile-copy-recursive-perl openbsd-inetd update-inetd
Os NOVOS pacotes a seguir serão instalados:
   libfile-copy-recursive-perl openbsd-inetd telnetd up-
date-inetd
0 pacotes atualizados, 4 pacotes novos instalados, 0 a
serem removidos e 2 não atualizados.
É preciso baixar 124 kB de arquivos.
Depois desta operação, 475 kB adicionais de espaço em
disco serão usados.
Você quer continuar [S/n]? S
Obter:1  http://debian.pop-sc.rnp.br/debian/  squeeze/
main libfile-copy-recursive-perl all 0.38-1 [20,6 kB]
Obter:2  http://debian.pop-sc.rnp.br/debian/  squeeze/
main update-inetd all 4.38+nmu1 [20,6 kB]
Obter:3  http://debian.pop-sc.rnp.br/debian/  squeeze/
main openbsd-inetd amd64 0.20080125-6 [37,8 kB]
Obter:4  http://debian.pop-sc.rnp.br/debian/  squeeze/
main telnetd amd64 0.17-36 [44,6 kB]
Baixados 124 kB em 0s (2122 kB/s)
Pré-configurando pacotes ...
Selecionando pacote previamente não selecionado libfile-
copy-recursive-perl.
(Lendo banco de dados ... 27532 ficheiros e directórios
actualmente instalados.)
Desempacotando libfile-copy-recursive-perl (de .../lib-
file-copy-recursive-perl_0.38-1_all.deb) ...
Selecionando pacote previamente não selecionado update-
inetd.
```

```
  Desempacotando update-inetd (de .../update-
inetd_4.38+nmu1_all.deb) ...
  Selecionando pacote previamente não selecionado open-
bsd-inetd.
  Desempacotando openbsd-inetd (de .../openbsd-in-
etd_0.20080125-6_amd64.deb) ...
  Selecionando pacote previamente não selecionado tel-
netd.
  Desempacotando telnetd (de .../telnetd_0.17-36_amd64.
deb) ...
  Processando gatilhos para man-db ...
  Configurando libfile-copy-recursive-perl (0.38-1) ...
  Configurando update-inetd (4.38+nmu1) ...
  Configurando openbsd-inetd (0.20080125-6) ...
  Stopping internet superserver: inetd.
  Not starting internet superserver: no services enabled.
  Configurando telnetd (0.17-36) ...
  Adicionando usuário telnetd ao grupo utmp
```

Após a instalação do servidor *Telnet*, o servidor estará "escutando" conexões na porta 23 (TCP), aguardando por solicitações de *login* no sistema. Por padrão, o *Telnet* realizará a autenticação usando os usuários do sistema que estão resididos no arquivo "/etc/passwd" e utilizará o mecanismo padrão de autenticação da máquina, baseado em PAM.

A seguir, você pode ver um exemplo de tentativa de conexão a um *host* utilizando *Telnet*.

```
# telnet 192.168.1.43
```

O *Telnet* caiu em desuso basicamente por razões de segurança, requisito exigido atualmente para Internet e Intranet, que são sistemas baseados em redes. Sua principal falha de segurança é o não suporte à criptografia, ou seja, todos os dados, desde o *login* do usuário até os comandos e operações realizadas na sessão, trafegam em texto-plano (*plain-text*) na Internet, podendo, desta forma, serem capturadas e utilizadas de má fé.

FIQUE ALERTA

Para uma maior segurança na sua organização, evite ao máximo utilizar o *Telnet* em sua infraestrutura. Ela pode vir a se tornar uma grande falha de segurança.

RECAPITULANDO

Neste capítulo, você conferiu os protocolos de acesso remoto para servidores e clientes. Acompanhou uma discussão sobre o RDP (*Remote Desktop Protocol*), que é um padrão instituído pela Microsoft, e viu o VNC (*Virtual Network Computing*) que é um serviço que nos possibilita vermos a tela do computador remoto. Também conheceu o SSH e o *Telnet*, que são dois protocolos de acesso remoto a servidores, e agora já sabe o motivo do surgimento do SSH e por que o *Telnet* caiu em desuso. No próximo capítulo, conheceremos os servidores de diretórios de redes. Preparado? Então, vamos em frente!

Servidor de Diretórios de Rede

11

Neste capítulo, você conhecerá os aspectos relacionados aos servidores de redes, algumas implementações de serviços de diretórios e, ainda, dois exemplos de implantações de servidores de diretórios. Um baseado no *Active Directory* (AD) da Microsoft e outro baseado no OpenLDAP, que é um *software* de código-aberto muito utilizado para implantação de serviço de diretórios.

Ao final deste capítulo, você terá subsídios para:

a) conhecer os aspectos dos serviços de diretórios de rede;

b) entender as implementações dos servidores de diretórios;

c) conhecer os protocolos envolvidos;

d) conhecer o serviço de diretórios *Active Directory*; e

e) entender o funcionamento do serviço de diretórios OpenLDAP.

Continuando a viagem sobre os serviços de redes, siga em frente agora e descubra para que servem os servidores de diretórios de rede. Boa viagem!

11.1 ASPECTOS

Por definição, o servidor de diretório de redes, também chamado de serviço de diretórios, é um sistema que tem a responsabilidade de armazenar, organizar e prover acesso às informações que estão em diretórios. Estes diretórios não são as pastas (*directory*) que estamos acostumados a trabalhar, eles são mapeamentos entre nomes e valores. Podemos usar um exemplo de um catálogo de listas telefônicas, onde um nome tem um valor a ele atribuído, neste caso, um número de telefone.

Um serviço de diretório define um esquema de espaço de nomes (*namespace*) para as aplicações e redes. Desta forma, um *namespace*, neste contexto, é o termo usado para definir um ou mais objetos na árvore de diretórios. Os diretórios são desenhados para ter um conjunto de regras (*rules*) que determinam como os recursos de rede serão nomeados e identificados no ambiente. São estas regras que garantem que os nomes serão únicos e sem ambiguidades.

Com este fim, é usado um padrão para serviço de diretórios, chamado X.500, composto por uma série de padrões que cobrem os serviços de diretórios e são utilizados para o desenvolvimento de aplicações que desejam interoperar com serviços de diretórios. Ele foi desenvolvido pelo ITU-T (*International Telecommunication Union - Telecommunication Standardization Sector*), em conjunto com o CCITT, e foi aprovado para iniciar a disseminação do padrão, em 1988.

O padrão X.500 tem pelo menos 4 protocolos que foram definidos, usando os padrões por ele sugeridos, sendo eles o *Directory Access Protocol* (DAP), o *Directory System Protocol* (DSP), o *Directory Information Shadowing Protocol* (DISP) e o *Directory Operational Bindings Management Protocol* (DOP). O diretório principal é o DAP, que era um protocolo chamado de *heavy-weight* que implementavatoda a pilha de rede sugerida pela OSI.

Desta forma, muitas outras alternativas ao DAP foram desenvolvidas para permitir que clientes da Internet acessassem os serviços de diretório X.500, usando a pilha de rede TCP/IP. Dentre as muitas alternativas desenvolvidas, a mais conhecida é o *Lightweight Directory Access Protocol* ou LDAP, como é conhecido. Nos últimos anos, devido a estas alternativas que foram desenvolvidas, os DAP podem usar o TCP/IP para serem acessados, porém o LDAP se popularizou tanto que ele continua sendo muito utilizado como protocolo de acesso a serviço de diretórios.

fotógrafo

Figura 25 - Logo do OpenLDAP

Ainda sobre os *namespaces*, tanto no LDAP como no DAP (X.500) eles são chamados de nomes distintos ou *distinguished name* (DN) e eles são usados para fazer a relação de nomes únicos de objetos dentro do diretório. São eles que garantem que um nome e um ou mais atributos associados a este nome será único em toda a árvore, evitando assim inconsistência ou falhas na operação.

Mais adiante você verá dois exemplos de instalação de serviços de diretórios: a *Active Directory* (AD) da Microsoft, e o *OpenLDAP* que é um *software* de código aberto que implementa o LDAP para sistemas operacionais baseados em *Unix-like*, tais como Linux, Mac OS X, AIX, HP-UX, BSDs, entre outros.

Existem várias aplicações que implementam os protocolos e serviços. Desta forma, cabe ao administrador do sistema, efetuar uma análise dos requisitos que são desejados e determinar qual a melhor solução para seu problema. Confira, no quadro a seguir, as implementações de serviços e diretórios.

SERVIÇO	DESCRIÇÃO
Network Information Service	O serviço de informações de rede (NIS) foi desenvolvido pela Sun Microsystems como um serviço de diretórios para sistemas UNIX.
Active Directory	O *Active Directory* é um serviço de diretórios que foi desenvolvido pela Microsoft e usou como padrão o X.500. Ele foi criado para ser usando no *Microsoft Exchange Server*, mas com seu sucesso atualmente é usado para muitas ações em redes Windows.
eDirectory	O *eDirectory* foi a implementação da Novell para serviços de diretórios. A grande vantagem do *eDirectory* é sua interoperabilidade com múltiplas plataform as, tais como Linux, Windows e NetWare.
ApacheDS	O Apache Directory Service é o serviço de diretórios da *Apache Software Foundation*.
Open Directory	O Open Directory é usado em servidores Mac OS X e ele implementa o LDAP usando como base uma versão do OpenLDAP e integrando com autenticações SASL e Kerberos.
OpenLDAP	O *OpenLDAP* atualmente é referência na área de serviço de diretórios para sistemas operacionais, pois suporta múltiplas arquiteturas (inclusive Windows) e é completamente alinhado com as definições do protocolo LDAP.

Quadro 11 - Implementações de Serviços de Diretórios

Agora que você já conhece os aspectos dos serviços de diretórios e as implementações, confira, nos próximos itens, os *Softwares Active Directory* e o *OpenLDAP*.

11.2 *ACTIVE DIRECTORY*

O *Active Directory*, ou somente AD como é bem conhecido, é um *software* desenvolvido pela Microsoft para trabalhar sobre a plataforma Windows e sua primeira implementação existe desde o *Windows 2000 Server*. Ele surgiu da necessidade de centralização das informações, onde as organizações estavam buscando meios de criar domínios, a fim de que vários servidores e estações de trabalho pudessem trabalhar em um único domínio de operação, facilitando, desse modo, a centralização de autenticações, resoluções de nomes e compartilhamento de arquivos.

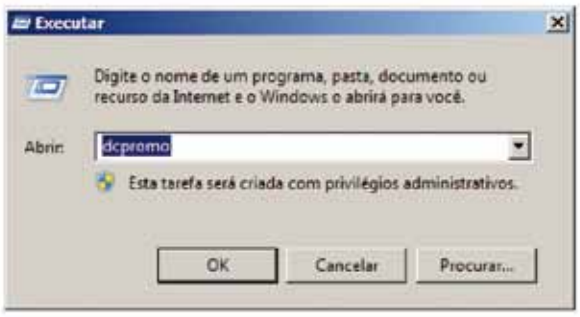
fotógrafo

Figura 26 - Logo do *Windows Server Active Directory*

O AD tem algumas características básicas de funcionalidade, como a centralização das informações sobre usuários, senhas e grupos de trabalho. O AD provê, aos sistemas operacionais e aplicações, facilidades para autenticação dos usuários. Além disso, permite a criação de usuários com permissões de acesso customizadas e criar sub-domínios, também conhecidos como as Unidades Organizacionais, ou como é chamado em inglês *Organization Units*.

As Organization Units podem ser encaradas como subdivisões do domínio principal. A necessidade deste elemento surgiu pelo fato de que gerenciar um domínio pode ser um desafio gerencial. Desta forma, fica mais fácil dividir este domínio principal em parte e, assim, restringir os acessos aos recursos da rede. Como exemplo, podemos citar um ambiente empresarial, onde podemos subdividir a empresa em áreas, cada uma com as devidas permissões de acesso, porém as máquinas e *logins* dos gerentes têm acesso a todas as áreas, controlando mais facilmente o ambiente.

Para efetuar a instalação e configuração deste servidor de diretórios, é necessário termos um sistema operacional Windows 2000 Server, Windows 2003 Server ou ainda o Windows 2008 Server, sendo qualquer das sub-versões destes sistemas compatível com o AD. Desta forma, o AD já vem pré-instalado nestes sistemas operacionais, bastando apenas fazer o *setup* da aplicação. Nos Windows citados, nós podemos dar início ao processo de instalação por meio da ferramenta "dcpromo.exe", que pode ser acessada via Menu Iniciar, e depois em Executar, digitando: "dcpromo", como a figura a seguir.



fotógrafo

Figura 27 - Execução do "dcpromo.exe"

Após a execução do comando, o sistema começará o processo de instalação da ferramenta. Dependendo do *hardware* do servidor em questão, isso pode levar alguns minutos. Após a etapa de processamento, o sistema de instalação dará início como pode ser visto na figura abaixo. Neste processo de instalação, você poderá escolher todas as configurações do seu AD, ou ainda, ir em um modo mais superficial para apenas dar início ao diretório. A informação fundamental que ele pedirá no processo é o FQDN (*Full Qualified Domain Name*), como por exemplo: ad.exemplo.com.

fotógrafo

Figura 28 - Tela inicial de configuração do AD

SAIBA MAIS

Para saber mais sobre a implementação de servidores de diretórios *Active Directory*, acesse a página: <http://www.microsoft.com/en-us/server-cloud/windows-server/active--directory-overview.aspx>.

Outras informações serão questionadas, como por exemplo, se você quer tornar seu AD o servidor DNS de sua rede. Se você não tiver um servidor DNS, é uma boa ideia aceitar para, desta forma, começar a identificar os computadores de sua rede, a partir do nome e não do IPs, como visto no capítulo de servidores DNS.

Depois do processo de instalação, será exigido que se reinicie o servidor para que o AD finalize o processo de instalação. Quando o servidor terminar esta etapa, as aplicações que fazem a interface com o AD podem ser acessadas no menu Iniciar, depois em Ferramentas Administrativas, como ilustrado na figura a seguir.

fotógrafo

Figura 29 - Menu de Ferramentas Administrativas

FIQUE ALERTA

A estruturação de um serviço de diretórios AD nas organizações, por muitas vezes, é feita sem controle e análise do que se espera da ferramenta, por esta razão, atrapalhando mais do que ajudando. Por isso, é muito importante que ocorra uma análise profunda da organização da rede e seja feito um levantamento de todos os requisitos exigidos e, realmente necessários na empresa, para que a solução venha a agregar, não, a atrapalhar. Então, fique alerta!!!

É muito importante que se analise todas as funcionalidades do AD, que explore cada um dos itens que compõem esta grande ferramenta e que, por fim, se alinhe a aplicação ou não destas tecnologias para atender os requisitos que são esperados da ferramenta. A implementação do *Active Directory* em redes Windows é muito importante na área de gerência da rede. A centralização de operações que o AD proporciona é vital para a boa saúde da rede e dos administradores (por que não?), desta forma é muito importante se estudar mais detalhadamente esta ferramenta para explorar todo seu potencial.

Agora, acompanhe o Casos e relatos a seguir, e associe o exemplo com o que você acabou de estudar para compreender melhor o assunto.

CASOS E RELATOS

Organização com diretórios

Uma grande empresa metalúrgica estava com problemas de integração ente os seus sistemas. A empresa já tinha mais de 30 anos de história mas, começou a investir fortemente em tecnologia há 5 anos atrás. Dessa forma, muitos sistemas foram postos para funcionar nos variados setores, como RH, Financeiro, Relações com o Mercado, Vendas, etc. A empresa contava, também, com outros sistemas para apoiar o negócio, como e-mail, servidores de arquivos, entre outros. Como a maioria desses sistemas não tinha integração alguma, isso dificultava muito o trabalho da área de TI. Sempre que um usuário de Vendas, por exemplo, tinha pelo menos 4 *logins* e senhas para acessar os sistemas, seu *e-mail* pessoal, o servidor de arquivos e o sistema financeiro. A gerência deste ambiente estava se tornando um problema, então, os responsáveis pela administração da rede foram chamados para discutirem e encontrarem uma possível solução. A solução sugerida pelos administradores foi a implantação de um serviço de diretórios onde seriam concentrados todos os usuários e senhas. Dessa forma, eles colocariam todos os sistemas efetuando a autenticação no serviço de diretórios. Implantaram, então, o *Active Directory*, distribuída da seguinte forma: gerência, recursos humanos, vendas, financeiro, operacional, relações com o mercado, outros. Cada um destes grupos recebeu acesso restrito a determinados perfis das aplicações e direitos de acesso aos serviços.

O problema enfrentado, após a migração de todos os dados de autenticação para o servidor, foi negociar com as empresas terceirizadas que desenvolviam os sistemas de Vendas e Financeiro e adaptarem a aplicação para suportarem autenticação baseada em diretórios. Ao final de três meses, o processo foi concluído com sucesso e, a partir de agora, cada funcionário independentemente do cargo ou função, tinha apenas um *login* e senha de acesso aos sistemas e serviços de rede, facilitando a gerência do ambiente e aumentando os níveis de segurança.

Você acabou de conhecer e conferir um exemplo de uso do *Activity Directory*. Agora, conheça co OpenLDAP e veja as suas qualidades.

11.3 OPENLDAP

O projeto OpenLDAP iniciou, em 1998, com o desenvolvedor Kurt Zeilenga, com o objetivo de ser uma solução mais "leve" para implementação do protocolo LDAP. O OpenLDAP é um *software* livre que implementa o *Lightweight Directory Access Protoco*. Ele suporta uma gama de plataformas, entre elas BSDs, AIX, HP--UX, Mac OS X, Solaris, Microsoft Windows, Linux, Android.

O *software OpenLDAP* funciona com três componentes básicos: o *slapd*, que é o servidor (*daemon*) que controla a operação de acesso a diretórios, um conjunto de bibliotecas que proporcionam acesso e, por fim, os *softwares* clientes,como o *ldapsearch, ldapadd, ldapdelete, slapcat*, entre outros.

Sob o ponto de vista arquitetural do OpenLDAP, ele foi dividido no início de seu desenvolvimento em duas partes: a primeira, uma interface com o cliente que trabalha com o acesso aos diretórios usando o protocolo LDAP e a segunda, um *backend*, que pode ser entendido como um banco de dados de objetos do diretório e tem a função de armazenar os objetos do diretório. O OpenLDAP suporta uma grande quantidade de *backends* sendo o mais popular e usado o BerkeleyDB.

VOCÊ SABIA?

Desenvolvedores estão tentando colocar bancos de dados relacionais, tais como: *PostgreSQL* e *MySQL* como *backend* do *OpenLDAP*.

De forma geral, o modo de operação do *OpenLDAP* recebe uma solicitação de um cliente no *frontend* (*slapd*), decodifica esta solicitação e, na sequência, envia para o *backend* (*BerkeleyDB*) processar. Quando o *backend* terminar o processamento, a solicitação retornará o resultado para o *frontend*, que enviará o resultado para o cliente LDAP solicitante. Basicamente é o modo de operação cliente-servidor clássico.

Confira a instalação do *OpenLDAP* em um sistema operacional Debian GNU/Linux na versão *Squeeze*. A seguir, você pode ver a instalação do *slapd*. É importante salientar que no processo de instalação, o instalador só irá questionar uma senha do administrador do *OpenLDAP*.

```
  root@server:/home/douglas# apt-get install slapd ldap-
utils
  Reading package lists... Done
  Building dependency tree
  Reading state information... Done
  The following NEW packages will be installed:
    ldap-utils slapd
  0 upgraded, 2 newly installed, 0 to remove and 29 not
upgraded.
  Need to get 0 B/1,917 kB of archives.
  After this operation, 4,678 kB of additional disk space
will be used.
  Preconfiguring packages ...
  Selecting previously deselected package slapd.
  (Reading database ... 195323 files and directories cur-
rently installed.)
  Unpacking slapd (from .../slapd_2.4.23-7.2_amd64.deb)
...
  Selecting previously deselected package ldap-utils.
  Unpacking ldap-utils (from .../ldap-utils_2.4.23-7.2_
amd64.deb) ...
  Processing triggers for man-db ...
  Setting up slapd (2.4.23-7.2) ...
    Moving old database directory to /var/backups:
    - directory unknown... done.
    Creating initial configuration... done.
    Creating LDAP directory... done.
  Starting OpenLDAP: slapd.
  Setting up ldap-utils (2.4.23-7.2) ...
```

O servidor LDAP está instalado, agora vamos para a etapa de configuração da ferramenta. Por padrão, ela vem vinculada com o domínio que está configurado na máquina. Desta forma, é altamente aconselhado que se reconfigure à aplicação para atender todos os requisitos que se esperam. Em sistemas operacionais, baseados em Debian, podemos executar a reconfiguração com o comando listado a seguir.

```
root@molar:/etc/ldap# dpkg-reconfigure slapd
Stopping OpenLDAP: slapd.
  Moving old database directory to /var/backups:
  - directory unknown... done.
  Creating initial configuration... done.
  Creating LDAP directory... done.
  Enabling LDAPv2 support... done
Starting OpenLDAP: slapd.
root@molar:/etc/ldap#
```

No processo de reconfiguração, serão questionadas algumas informações sobre o *backend* a ser utilizado (BDB ou HDB) e suporte a outras versões do LDAP, mas a informação mais importante é sobre o domínio da máquina, que formará a base de sua árvore LDAP. Vamos usar o domínio exemplo.com, por isso a base do nosso LDAP esperada deve ser: dc=exemplo,dc=com. Para confirmar se nossa base está corretamente configurada, podemos usar o comando *slapcat*, que tem a função de exigir o conteúdo da base. Confira a saída do comando *slapcat*.

```
root@molar:/etc/ldap# slapcat
dn: dc=exemplo,dc=com
objectClass: top
objectClass: dcObject
objectClass: organization
o: exemplo.com
dc: exemplo
structuralObjectClass: organization
entryUUID: 2decc3b6-7e3f-1030-8dce-cda9c79239e5
creatorsName: cn=admin,dc=exemplo,dc=com
createTimestamp: 20110928170057Z
```

```
entryCSN: 20110928170057.564210Z#000000#000#000000
modifiersName: cn=admin,dc=exemplo,dc=com
modifyTimestamp: 20110928170057Z

dn: cn=admin,dc=exemplo,dc=com
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator
userPassword:: e1NTSEF9Sk9Zc0taRXRwNUJ6WGxnZkRheldYUm-
JJK2h0VElydi8=
structuralObjectClass: organizationalRole
entryUUID: 2ded35f8-7e3f-1030-8dcf-cda9c79239e5
creatorsName: cn=admin,dc=exemplo,dc=com
createTimestamp: 20110928170057Z
entryCSN: 20110928170057.567143Z#000000#000#000000
modifiersName: cn=admin,dc=exemplo,dc=com
modifyTimestamp: 20110928170057Z
```

Com esta etapa cumprida, podemos continuar a configuração do nosso servidor LDAP. A primeira etapa é gerar novamente a senha de administrador para termos o *hash* para inserirmos no nosso arquivo de configuração principal, o "/etc/ldap/slapd.conf". Feito isto, podemos criar, agora, o arquivo de configuração como mostrado, a seguir. Note que isto é apenas um exemplo funcional, para fins de estudo. Para servidores de produção, deve-se estudar a solução e adequá-la de acordo com os requisitos.

```
root@molar:/etc/ldap# cat slapd.conf
# slapd.conf

include         /etc/ldap/schema/core.schema
include         /etc/ldap/schema/cosine.schema
include         /etc/ldap/schema/nis.schema
include         /etc/ldap/schema/inetorgperson.schema

allow        bind_v2
```

```
   pidfile         /var/run/slapd/slapd.pid
   argsfile     /var/run/slapd/slapd.args
   loglevel     none
   modulepath     /usr/lib/ldap
   moduleload     back_bdb

   sizelimit 500
   tool-threads 1

   backend         bdb
   database     bdb

   suffix         "dc=exemplo,dc=com"
   rootdn         "cn=admin,dc=exemplo,dc=com"
   rootpw          {SSHA}YZTARuiPLLtqW0TgrMkhPU0dnTZSB8+/

   directory    "/var/lib/ldap"
   dbconfig set_cachesize 0 2097152 0
   dbconfig set_lk_max_objects 1500
   dbconfig set_lk_max_locks 1500
   dbconfig set_lk_max_lockers 1500

   index         objectClass eq
   lastmod         on
   checkpoint     512 30

   access     to     attrs=userPassword,shadowLastChange
      by     dn="cn=admin,dc=leonardoamorim,dc=com,dc=br"
write
      by anonymous auth
      by self write
      by * none

   access to dn.base="" by * read

   access to *
      by dn="cn=admin,dc=exemplo,dc=com" write
      by * read
   root@molar:/etc/ldap#
```

Após a criação e customização do principal arquivo de configuração do *OpenLDAP*, é preciso gerar novamente a configuração do *OpenLDAP*, para sincronizá-lo com as novas diretrizes que foram definidas no arquivo de configuração. Isto pode ser feito com a sequência de comando listada abaixo.

a) Primeiro você deve parar o servidor LDAP: **#/etc/init.d/slapd stop**.Depois, acesse o diretório do OpenLDAP: **# cd /etc/ldap**.

b) Faça uma cópia de segurança do diretório de configuração do servidor: **# cp –R slapd.d slapd.d.BKP**.

c) Agora, apague o diretório de configuração, e crie um novo, vazio: **# rm -rf slapd.d** e **# mkdir slapd.d.**

d) Agora use o comando "slaptest" para ler o arquivod de configuração e gerar a nova configuração: **# slaptest -f slapd.conf -F slapd.d.**

e) Altere as permissões dos arquivos que foram gerados, para que o servidor possa manipulá-los: **# chown -R openldap:openldap slapd.***

f) Com tudo realizado, basta iniciar o servidor novamente: **# /etc/init.d/slapd start.**

Realizadas estas configurações, o servidor LDAP estará pronto para receber estruturas em seus diretórios. Ele estará escutando na porta 389 (TCP) do servidor e pronto para responder solicitações LDAP. Sobre o acesso e interação com a base, ela pode ser feita via linha de comando e via interfaces gráficas. Sobre a linha de comando, o quadro, a seguir, apresenta alguns comandos e suas descrições. Observe.

COMANDO	DESCRIÇÃO
slappasswd	Gera senhas para o administrador ou usuários do LDAP.
ldapadd	Adiciona entradas no diretório LDAP.
ldapdelete	Deleta entradas do diretório LDAP.
ldapmodify	Modifica entradas existentes no diretório LDAP.
ldappassword	Modifica entradas de senhas do diretório LDAP.
ldapsearch	Procura entradas no diretório LDAP.
slapcat	Exporta um arquivo diretamente do banco de dados LDAP. Geralmente é exportado no formato LDIF.
slaptest	Verifica o arquivo "slapd.conf".
slapd	É o daemon (servidor) LDAP.
slapadd	Adiciona entradas LDAP no banco de dados.
slapauth	Gerencia autenticações.
slapdn	Verifica o Distinguished Name no DIT (Directory Information Tree ou Árvore de Informações de Diretórios).

Quadro 12 - Comandos e descrições para interação com *OpenLDAP*

SAIBA MAIS

Saiba mais informações sobre modelos de implementação, funcionalidades e caraterísticas do OpenLDAP no *site* oficial da ferramenta em: <http://www.openldap.org>.

Como você já sabe, é possível fazer a interação com o OpenLDAP por meio de interfaces gráficas. Alguns administradores gostam mais deste aspecto visual para gerenciar seus serviços, porém é muito importante que antes de usar interfaces para facilitar o serviço, o administrador saiba operar o serviço via linha de comando. As interfaces mais utilizadas para gerência de diretórios OpenLDAP são o "phpldapadmin" e o "LDAPadmin". Sendo a primeira delas baseada em servidores Web e a segunda, em uma aplicação *stand-alone* (do tipo que é instalada na máquina do cliente) que funciona em Windows. Na figura a seguir, você pode ver a interface do "phpldapadmin".

fotógrafo

Figura 30 - Interface do phpLDAPadmin

SAIBA MAIS

Você encontra mais informações sobre as funcionalidades, modo de operação e instalação do phpLDAPadmin no *site* oficial da ferramenta: <http://phpldapadmin.sf.net>.

Por fim, é importante ressaltar que o *OpenLDAP* sozinho não é tão útil como ele integrado com outras ferramentas. De nada serve um diretório se ele não estiver guardando e servindo informações para aplicações. Esta integração, na maioria dos casos, é feita por meio de esquemas (*schemes*) que são adicionados na configuração do *OpenLDAP* (*slapd.conf*) e estendem as funcionalidades do diretório, podendo assim integrar-se com outras ferramentas, tais como *Samba*, *Postfix*, *Bind9*, *Squid*, entre outras ferramentas.

Como visto, o *OpenLDAP* é uma excelente opção para construção de serviços de diretórios, pois ele une a flexibilidade dos *softwares* livres, com uma implementação leve de um protocolo muito utilizado para centralização de informações, principalmente de autenticação de serviços em redes de computadores.

Como você já sabe, é necessário muito estudo antes de qualquer implementação do *OpenLDAP*. O administrador deve analisar o cenário da rede, verificar os requisitos que devem ser atendidos, para que a implementação do serviço seja um sucesso.

RECAPITULANDO

Neste capítulo, você conheceu os aspectos relacionados a servidores de diretórios que também são conhecidos como serviço de diretórios. Foram vistos, neste estudo, as características dos serviços de diretórios e seus componentes. Viu ainda o *Active Directory* da Microsoft, que é um serviço de diretórios para sistemas baseados em Windows e o *OpenLDAP*, que é um servidor de diretórios para sistemas baseados em UNIX. No próximo capítulo, você entenderá como são os serviços de transferência de arquivos. Até lá!

Servidor de Transferência de Arquivos

12

Neste capítulo, você conhecerá aspectos relacionados a servidores de transferência de arquivos. Verá um pouco da história dos protocolos e suas respectivas RFCs que definem os padrões, estudará o protocolo em si e seus dois modos de funcionamento (ativo e passivo), as ferramentas para implementação de servidores FTP e TFTP e, por fim, poderá conferir a instalação de um servidor de transferência de arquivos, utilizando o "vsftpd" em um sistema operacional *Debian* Linux/GNU versão *Squeeze*.

Ao final deste capítulo, você terá subsídios para:

a) conhecer a evolução e a história dos servidores de transferência de arquivos;

b) entender o protocolo FTP;

c) compreender o modo de funcionamento do FTP;

d) entender o protocolo TFTP; e

e) conhecer as ferramentas para implementação de servidores FTP e TFTP.

Bem vindo a mais uma parada dessa viagem ao mundo dos serviços de redes. Ajeite-se confortavelmente, aperte o cinto e siga em frente!

12.1 HISTÓRIA

Para a transferência de arquivos na Internet, o padrão adotado, até hoje, é o FTP (*File Transfer Protocol*), que é construído no paradigma cliente-servidor e utiliza diferentes canais de comunicação para controle e fluxo de dados. Outro tipo de protocolo utilizado para transferir arquivos, é o TFTP (*Trivial File Transfer Protocol)*. Este é um protocolo conhecido por sua simplicidade e pouca flexibilidade. Ele é geralmente utilizado para transferir arquivos de configuração, como por exemplo, IOS de *switchs* e roteadores, ou ainda, transferir arquivos de inicialização (*boot*) entre máquina em um ambiente local.

O FTP foi definido na RFC 114, no ano de 1971, e sofreu alterações nos anos posteriores, até 1975. As primeiras versões de aplicações que implementavam este protocolo rodavam no MIT (*Massachusetts Institute of Technology*). Em 1980, um pesquisador chamado Jon Postel escreveu a RFC 765 que descrevia o FTP rodando sobre o topo da pilha de protocolos TCP. Atualmente a RFC 959 é a referência oficial que determina seus padrões.

O protocolo TFTP (*Trivial File Transfer Protocol*) foi definido, em 1980, e atualmente é normatizado pela RFC 1350. Após esta data, algumas extensões ao protocolo foram sugeridas e estão documentadas em outras RFCs. Como exemplos destas extensões, podemos citar a modificação de tamanho máximo de arquivo a ser transferido que, em 1998, foi estendido de 32Mb para 4Gb e está normatizado na RFC 2347.

Conheça um pouco melhor o protocolo FTP no item a seguir.

12.2 O PROTOCOLO FTP

O protocolo FTP pode ser considerado um dos precursores quando o objetivo é transferir arquivos em redes de computadores. Em 1971, na sua primeira definição, os requisitos tecnológicos e de segurança eram outros, que hoje não se aplicam. Atualmente a utilização de um protocolo de transferência de arquivos, que não proporcione suporte a criptografia, não é cogitado como padrão em qualquer operação que envolva criticidade dos dados.

O FTP trabalha no topo da pilha de protocolos TCP/IP (camada de aplicação), diferentemente dos protocolos NTP, LDAP e DNS. Ele trabalha em duas portas simultaneamente, sendo a porta 21 conhecida como “canal de controle”, e a porta 20, conhecida como “canal de dados”. Como padrão, o protocolo FTP usa a porta 21 para controlar a conexão, mas a conexão de dados deve ser determinada pelo método que o cliente solicita ao servidor, que pode ser ativo ou passivo.

FIQUE ALERTA

Por padrão, o protocolo FTP não tem criptografia e seus dados trafegam livremente em texto-plano. Então, fique alerta!

O FTP apresenta dois métodos de funcionamento. Confira, no item a seguir, quais são esses dois métodos.

12.3 MODOS DE FUNCIONAMENTO DO FTP

Existem dois métodos de funcionamento para protocolos FTP: o **ativo** e o **passivo**. As conexões FTP de modo ativo são algumas vezes chamadas de conexões "gerenciadas pelo cliente" porque o cliente envia um comando PORT ao servidor na conexão do controle. O comando solicita ao servidor que estabeleça uma conexão de dados da porta TCP 20 no servidor até o cliente com a porta TCP especificada pelo comando PORT.

As conexões FTP de modo passivo são, às vezes, chamadas de conexões "gerenciadas pelo servidor" porque, depois que o cliente emite o comando PASV, o servidor responde com uma de suas portas temporárias usadas como a porta do servidor na conexão de dados. Depois que um comando de conexão de dados é emitido pelo cliente, o servidor se conecta ao cliente, usando a porta imediatamente acima da porta do cliente na conexão do controle.

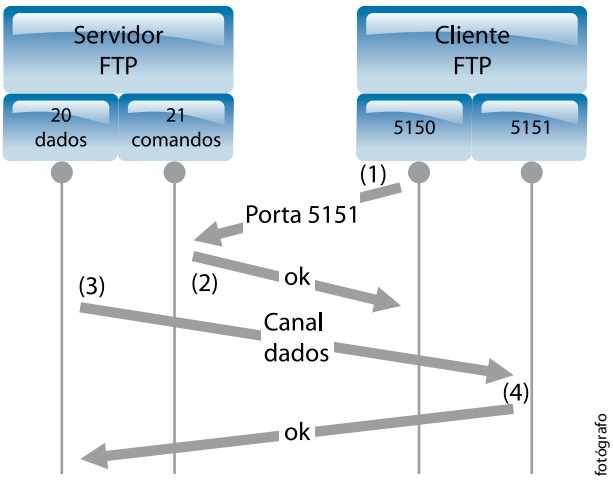


Figura 31 - Transferência em Modo Ativo

Conforme podemos visualizar na figura, em uma conexão FTP comum o cliente indica, aleatoriamente, a porta em que deseja que os dados sejam transferidos. No exemplo dessa figura, esta porta é a 5151. Assim, seguindo a numeração indicada na figura, quando o cliente acessa o serviço de FTP (porta 21) de um servidor, ele indica para qual porta o servidor deverá lhe enviar os dados (no exemplo, esta é a porta 5151). Deste modo, quando o *download/upload* começar, o servidor enviará os dados (partindo da porta 20) para essa porta especificada pelo cliente (FTP-RNP, 2011).

O problema desse modo de funcionamento do FTP, é que não é possível, para o servidor, saber o número da porta que o cliente solicitará para o envio dos dados, o que obriga o administrador a deixar todas as portas TCP abertas no *firewall*. Uma maneira de se contornar isso, é abrir apenas as portas altas (> 1024). Mas, mesmo assim, esse cenário é muito propício às tentativas de ataques ou mesmo instalação de *backdoors*.

Isto acarreta um trabalho extra, na administração do servidor, como por exemplo, as atualizações de *patches* constantemente, correções de *bugs* e configurações extra de segurança, como por exemplo, a utilização de "cadeias" (*chroot*) para controlar o perímetro do servidor. Entretanto, vale salientar que este é o modo mais utilizado de servidores FTP, visto sua facilidade de instalação e controle sobre o serviço.

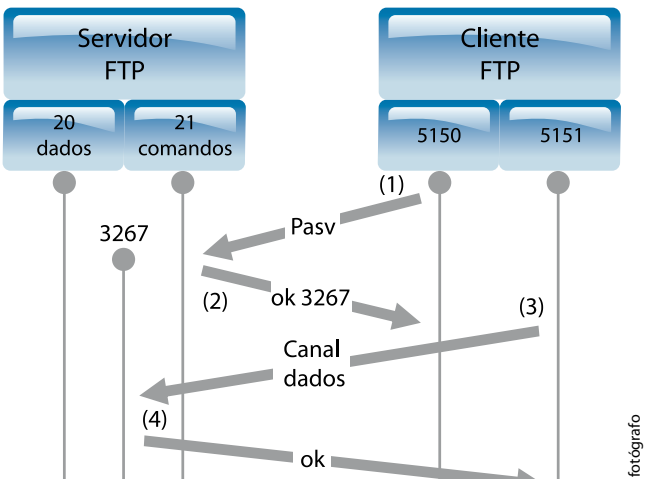


Figura 32 - Transferência em Modo Passivo

Como se pode ver, agora é o servidor que, no início da conexão, informa ao cliente por qual de suas portas TCP os dados serão enviados. Conforme sugere a orientação da seta, o início do processo de transferência é iniciado pelo cliente, ficando o servidor numa posição passiva. Acompanhando a numeração desta última figura, observamos que, quando o cliente acessa o serviço de FTP (porta 20) de um servidor, a conexão já é estabelecida como "PASV" (passiva). Desta forma, é o servidor quem indica por qual porta os dados serão trocados com o cliente (no exemplo esta é a porta 3267 (FTP-RNP, 2011).

Uma vez que as portas usadas na transferência de dados são definidas pelo servidor, pode-se também definir, no *firewall* (ou roteador), que apenas essas portas poderão ser acessadas de fora da rede. Deste modo, as tentativas de ataque restringem-se a um limite especificado pelo número de portas referentes ao serviço de FTP. Embora o processo envolvido na conexão FTP se altere significativamente, a operação é transparente para o usuário, a menos que ele utilize um browser FTP que não aceite o modo passivo. São poucos hoje em dia, os clientes FTP que não aceitam esse modo de comunicação com o servidor FTP.

A seguir, mais informações sobre o protocolo FTP. Continue atento e confira as informações do próximo item.

12.4 O PROTOCOLO TFTP

O Protocolo de Transferência de Arquivos Triviais ou *Trivial File Transfer Protocol* (TFTP) é um tipo de protocolo simples para transferência de arquivos. Ele é implementado em cima do UDP (*User Datagram Protocol*) usando a porta número 69 como padrão. Este protocolo foi projetado para ser pequeno e de fácil implementação, desta forma, não tem a maioria das características comuns de um servidor de transferência de arquivos (FTP). A sua função básica é ler e gravar arquivos a partir de um servidor, ou ainda para um servidor.

Neste protocolo, qualquer transferência começa com um pedido para ler ou escrever um arquivo, que também serve para solicitar uma conexão. Se o servidor conceder o pedido, a conexão é aberta e o arquivo é enviado em blocos de comprimento fixo de 512 *bytes*. Cada pacote de dados contém um bloco de dados e deve ser reconhecido por um pacote de confirmação, antes do próximo pacote ser enviado.

Se um pacote se perde na rede, o destinatário terá um tempo limite e pode retransmitir seu último pacote, forçando assim o remetente do pacote perdido a retransmitir o que foi perdido. O remetente tem que manter apenas um pacote para retransmissão. É importante salientar que ambas as máquinas envolvidas na transferência são consideradas remetentes e receptores. O TFTP normalmente usa UDP como seu protocolo de transporte, mas não é uma exigência.

O TFTP não tem nenhum mecanismo de segurança implementado por padrão, nem mesmo modos de autenticação são fornecidas pela especificação do protocolo. Ainda, outra limitação é ele trabalhar sobre o UDP, desta forma não tendo todo o *background* de segurança que o TCP fornece aos protocolos que ele trabalha. Assim, a utilização deste protocolo se torna, de certa forma, inviável para a maioria das operações de servidores de arquivos. Este protocolo é largamente utilizado como servidores de atualizações para *switchs* e roteadores de rede. Neste modo, alguns equipamentos somente suportam atualizações de *firmware* a partir de servidores TFTP.

VOCÊ SABIA?

O TFTP está caindo em desuso. Para CISCO e 3com, ele não é mais utilizado como servidor de atualização para equipamentos de rede. Seu substituto é o protocolo SSH e/ou FTP.

12.5 FERRAMENTAS PARA SERVIDORES FTP

Existem várias aplicações que implementam os servidores de transferência de arquivos FTP e TFTP. Confira, no quadro a seguir, uma lista de ferramentas e suas plataformas.

FERRAMENTA	PLATAFORMAS	PROTOCOLOS
Apache FTP Server	Windows, Linux, UNIX, Mac OS X	FTP
glFTPd	Linux, BSD, Mac OS X	FTP, FTPS
ProFTPD	Linux, BSD, Mac OS X	FTP
Pure-FTPd	Linux, BSD, Mac OS X	FTP
SlimFTPd	Windows	FTP
Vsftpd	Linux, BSD, UNIX	FTP, FTPS
Wu-ftpd	Linux, BSD, Solaris, Mac OS X, UNIX	FTP
atftp	Linux, BSD, UNIX	TFTP
tftp-server	Linux, UNIX	TFTP
TFTP-Server	Mac OS X	TFTP
TFTPD32	Windows	TFTP

Quadro 13 - Ferramentas para FTP e TFTP

CASOS E RELATOS

Popularização com troca de protocolos

Uma organização sem fins lucrativos da área de desenvolvimento de *software* tinha um problema claro: como transferir de forma eficiente e rápida as imagens de suas distribuições de sistemas operacionais para os usuários finais, uma vez que essas imagens têm tamanhos significativos, muitas vezes ultrapassando 7Gb? A solução inicial do administrador de redes da organização foi estruturar um esquema de transferência de imagens via protocolo Torrent. Só que, com essa solução, os usuários precisariam ter um cliente deste protocolo e baixar o arquivo ".torrent" da imagem do sistema operacional que desejariam usar para então fazer o download do arquivo. O problema dessa abordagem é que a maioria dos usuários tinha certa resistência na utilização do protocolo *torrent*, e isso acabou impactando diretamente na popularização do sistema operacional da organização. Coube então, ao administrador da rede, encontrar uma forma mais fácil de transferir estas imagens aos usuários. Após estudar o problema, o administrador resolveu utilizar o protocolo FTP e estruturar um servidor FTP público com acesso anônimo. Com essa solução, a empresa notou que os *downloads* de seus *softwares* aumentaram exponencilmente. Em uma pesquisa de satisfação feita posteriormente, 53% dos usuários marcou que a nova forma de *download* é um dos pontos fortes do sistema operacional da empresa. Foi uma solução simples, mas que resolveu um dos grandes problemas que a empresa tinha.

Viu só como o FTP pode ser uma forma simples de resolver grandes problemas? Acompanhe, agora, como é feita a instalação de um servidor FTP.

12.6 INSTALAÇÃO DE UM SERVIDOR FTP

Como exemplo de instalação de um servidor de transferência de arquivos, usaremos o *vsftpd* que, como os próprios desenvolvedores dizem, é o mais seguro e o mais rápido servidor FTP para sistemas Unix. Usaremos, como plataforma básica de instalação, um sistema operacional Debian Linux/GNU versão Squeeze.

A seguir, você pode visualizar a instalação da ferramenta utilizando o sistema gerenciador de pacotes "apt-get".

```
root@server:/# apt-get install vsftpd
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
Os NOVOS pacotes a seguir serão instalados:
  vsftpd
0 pacotes atualizados, 1 pacotes novos instalados, 0 a serem removidos e 2 não atualizados.
É preciso baixar 148 kB de arquivos.
Depois desta operação, 475 kB adicionais de espaço em disco serão usados.
Obter:1 http://debian.pop-sc.rnp.br/debian/ squeeze/main vsftpd amd64 2.3.2-3 [148 kB]
Baixados 148 kB em 0s (3407 kB/s)
Pré-configurando pacotes ...
Selecionando pacote previamente não selecionado vsftpd.
(Lendo banco de dados ... 27462 ficheiros e directórios actualmente instalados.)
Desempacotando vsftpd (de .../vsftpd_2.3.2-3_amd64.deb) ...
Processando gatilhos para man-db ...
Configurando vsftpd (2.3.2-3) ...
Starting FTP server: vsftpd.
```

VOCÊ SABIA?

É possível integrar o OpenSSL aos servidores FTP e, desta maneira, ter um canal de comunicação seguro, para transferência de dados, chamado SFTP.

O principal arquivo de configuração do servidor FTP “vsftpd” é o “vsftpd.conf” que, por padrão, está localizado no diretório “/etc”. Por padrão, ele vem configurado para suportar acessos anônimos e não suportar autenticação de usuários locais. Estará escutando na porta-padrão de controle (21). Este arquivo suporta parâmetros no formato “parâmetro = atributo”.

No quadro a seguir, pode-se visualizar algumas opções de configuração possíveis para o servidor FTP “vsftpd”.

PARÂMETRO	DESCRIÇÃO
anon_mkdir_write_enable	Se estiver definido como YES, usuários anônimos terrão permissão para criar novos diretórios sobre certas condições. Para isso funcionar, a opção **write_enable** deve estar ativa e o usuário de ftp anônimo deve ter permissão de escrita no diretório atual.
anonymous_enable	Controla se logins de usuários anônimos são permitidos ou não. Se habilitada, tanto o usuário **ftp** como o **anonymous** são reconhecidos como logins anônimos.
background	Quando habilitada e o vsftpd é iniciado em modo listen, o vsftpd colocará em background os processos de escuta, isto é, o controle retornará direto para a shell de execução.
dirmessage_enable	Se habilitado, usuários do servidor FTP podem receber mensagens quando eles entram pela primeira vez em um novo diretório. Por padrão, um diretório é escaneado pelo arquivo.message, mas você pode alterar isto com o ajuste de configuração **message_file**.
listen	Se ativado, o vsftpd executará em modo standalone. Isto significa que o vsftpd não deve ser executado por qualquer tipo de inetd. Ao contrário, o executável vsftpd é carregado diretamente. O próprio vsftpd irá, então, se preocupar em escutar as conexões recebidas.
local_enable	Controla se os logins locais são permitidos ou não. Se habilitado, usuários normais do /etc/passwd podem ser utilizados para logar no sistema.
ssl_enable	Se habilitado e o vsftpd estiver compilado com suporte a OpenSSL, o vsftpd irá suportar conexões seguras via SSL. Isto se aplica no controle de conexão (incluindo acessos) e também conexões de dados. Você precisará de um cliente com suporte à SSL também. ATENÇÃO! Cuidado ao habilitar esta opção. Apenas habilite se você necessita. Vsftpd não garante a segurança das bibliotecas do OpenSSL. Habilitando esta opção, você está declarando que você confia na segurança das bibliotecas instaladas do OpenSSL.
syslog_enable	Se habilitado, então qualquer registro direcionado para /var/log/vsftpd.log vai para o registro do sistema. Registros são feitos sob a facilidade FTPD.
secure_chroot_dir	Esta opção será o nome do diretório, quando o mesmo estiver vazio. Portanto, o diretório, não possuirá permissão de escrita pelo usuário FTP. Este diretório é usado com uma jaula chroot segura, quando o vsftpd não precisa de acesso ao sistema de arquivos.
banner_file	Esta opção aponta para o nome do arquivo que contém o texto que será visualizado quando algum usuário conectar ao servidor. Se ajustado, este substitui o texto visualizado pela opção ftpd_banner.

Quadro 14 - Parâmetros para o *vsftpd*

SAIBA MAIS

Para saber maiores informações sobre parâmetros opcionais e seus detalhamentos, vi*site* o *site* oficial da ferramenta: <vsftpd.beasts.org>.

Por fim, para testarmos nosso servidor FTP, temos que usar um cliente FTP para este fim. Para sistemas UNIX e Linux, em geral, temos o comando "ftp" na linha de comando, e com ele podemos avaliar e interagir com o nosso servidor. Para clientes Windows existe, também na linha de comando (Prompt-DOS), o comando "ftp" ou, ainda, é possível baixar algum *software* cliente para este fim.

A seguir, é possível ver um exemplo de interação com o servidor FTP, usando o cliente FTP "ftp" para Linux.

```
root@server:/# ftp ftp.exemplo.com
Connected to ftp.exemplo.com.
220 (vsFTPd 2.3.2)
Name (localhost:douglas): douglas
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -l
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
226 Directory send OK.
ftp> pwd
257 "/home/douglas"
ftp> exit
221 Goodbye.
```

RECAPITULANDO

Neste capítulo, você conheceu o FTP (*File Transfer Protocol*) e o TFTP (*Trivial File Transfer Protocol*). Ambos os protocolos são maduros e muito utilizados para transferência de arquivos, sendo o FTP mais utilizado por questões de flexibilidade e segurança, visto que o TFTP não tem controle de acesso nem de conexão, já que trabalha em UDP. Você conheceu, também, a história desses dois protocolos e o modo de funcionamento, sendo que o FTP pode ser utilizado de dois modos ou métodos de transferência de arquivo: ativo e passivo. Conheceu uma lista com algumas aplicações para implementação deste tipo de servidor em um exemplo de instalação baseado no servidor FTP *vsftpd* em cima de um sistema operacional Debian Linux/GNU na sua versão *Squeeze*. Para encerrar o capítulo, você acompanhou a configuração com suas respectivas descrições.

Até aqui, você já conheceu muita coisa sobre os servidores de vários serviços de rede, não é mesmo? Mas... ainda tem muita coisa para aprender. Outro assunto importante que você precisa conhecer são os serviços de sincronismo de relógio. Esse é o tema do próximo capítulo. Siga em frente, e bom estudo!

Servidor de Sincronismo de Relógio (NTP)

13

Neste capítulo, você conhecerá um pouco da história dos servidores de tempo e a necessidade que gerou o protocolo NTP, que atualmente está em sua versão 4 (NTPv4). Aprenderá sobre o modo de operação dos servidores de tempo baseados em servidores *stratum*, verá, ainda, os padrões de tempo GMT e UTC e acompanhará uma instalação básica de uma ferramenta para implantar um servidor de tempo.

Ao final desse capítulo, você terá subsídios para:

a) conhecer a evolução e a história dos servidores de sincronismo de relógios;

b) entender os modos de operação dos servidores de tempo;

c) conhecer os padrões de tempo;

d) conhecer os *softwares* para implementação de servidores de tempo.

13.1 HISTÓRIA

A cada novo computador que é instalado e configurado é necessário ajustar o relógio. Ainda mais, os relógios de computadores que deveriam, em velocidades diferentes, o que torna esta tarefa mais vital. Segundo Smith (2003), em uma grande rede de computadores, alguns dias após você ajustar os horários de todas as máquinas, o horários das mesmas já estarão desatualizados. Um dos implicadores nessas diferenças é o horário de verão a que estamos sujeitos todos os anos e que exige uma nova rodada de configurações em todas as máquinas.

Desta forma, manter os relógios dos computadores de uma rede sincronizados é uma tarefa árdua que, sem a ajuda de tecnologias que suportem isto, torna-se infindável. Para resolver esse problema, foram introduzidos os *Time Server*, (Servidores de Tempo) ou, ainda, os Servidores de Sincronismo de relógios, que são sinônimos. Eles nasceram com uma função simples, porém muito importante: manter relógios de uma rede sincronizados.

O protocolo NTP (*Network Time Protocol*) teve sua versão 3 normatizada pela RFC 1305, em 1992. Atualmente a versão de referência é a NTPv4 (versão 4) e este padrão é documentado na RFC 5905, de 2010. O protocolo NTP usa o algoritmo de Marzullo, desenvolvido por Keith Marzullo, em sua tese de doutorado, em 1984. O objetivo básico do algoritmo é reduzir o ruído e proporcionar uma qualidade de sincronização melhor ao protocolo NTP.

O protocolo NTP foi desenvolvido com objetivo de resistir aos efeitos de latência variável usando *buffer* de *jitter*. Com isto, o protocolo pode manter o tempo para o público na Internet na razão de 10 milissegundos e sobre o público em redes locais na razão de 1 milissegundo. Por padrão, o NTP usa o Tempo Universal Coordenador (UTC) como padrão de tempo.

VOCÊ SABIA?

Você sabia que o algoritmo do protocolo NTP ajusta a hora antes de entregá-la ao sistema operacional que requisitou? Isto é devido à latência das redes que pode variar de um lugar para outro.

Você conheceu o serviço de sincronismo de relógio. Sabia da existência desse tipo de serviço? Acompanhe, agora, como é feita a sua operação.

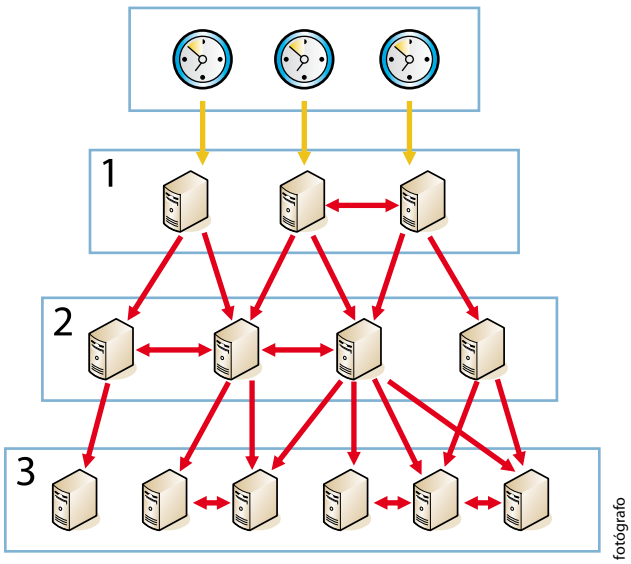
13.2 MODO DE OPERAÇÃO

Segundo Smith (2003), o modo de operação de um servidor NTP inicia com uma fonte de horário verídica, como um relógio atômico, um rádio ou um *modem* receptor que possa sincronizar com uma fonte oficial de horário. Os receptores de GPS (*Global Positioning Syste*m ou Sistema de Posicionamento Global) funcionam parcialmente utilizando sinais de horário de satélites, deste modo eles podem ser usados como uma forma de conseguir uma fonte de horário mais apurado.

O relógio atômico, dispositivo de rádio ou semelhante é também conhecido como servidor de referência ou um *stratum*. A menos que você esteja conectado diretamente a um destes dispositivos de referência, o servidor que você usa para se atualizar é, no máximo, um *stratum* 1.

Ainda segundo Smith (2003), em teoria, os *stratum* 1 são os servidores mais atualizados da Internet, visto que eles estão conectados diretamente ao *stratum* 0 (servidor de referência). Subsequentemente os servidores que se atualizam em servidores *stratum* 1, são chamados de *stratum* 2 e assim sucessivamente.

Na figura a seguir, é possível ver uma arquitetura básica de *stratum*.



fotógrafo

Figura 33 - Arquitetura Básica de uma Rede NTP

Nessa figura, é possível visualizar 3 servidores de referência (*stratum*), abaixo deles, 3 servidores de tempo (*stratum* 1), e logo abaixo seguem a arquitetura até *stratum* 3. É importante salientar que, quanto mais alto o *stratum*, na teoria, mais apurado será a hora fornecida.

A seguir, é possível visualizar uma figura completa da RNP, que ilustra uma arquitetura de servidores de tempo.

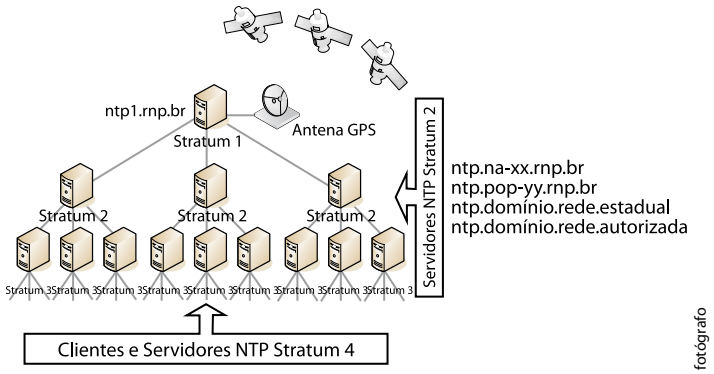


Figura 34 - Hierarquia de Servidores de Tempo
Fonte: RNP (2011)

Nessa figura, é possível ver, em uma primeira instância, na parte superior da imagem, uma antena GPS captando os sinais dos satélites e atualizando o stratum 1 da RNP. Depois entram os 3 servidores *stratum* 2 da RNP que fornecem hora pra uma grande quantidade de *stratum* 3 que, a seguir, atualizam relógios de clientes e servidores (*stratum* 4).

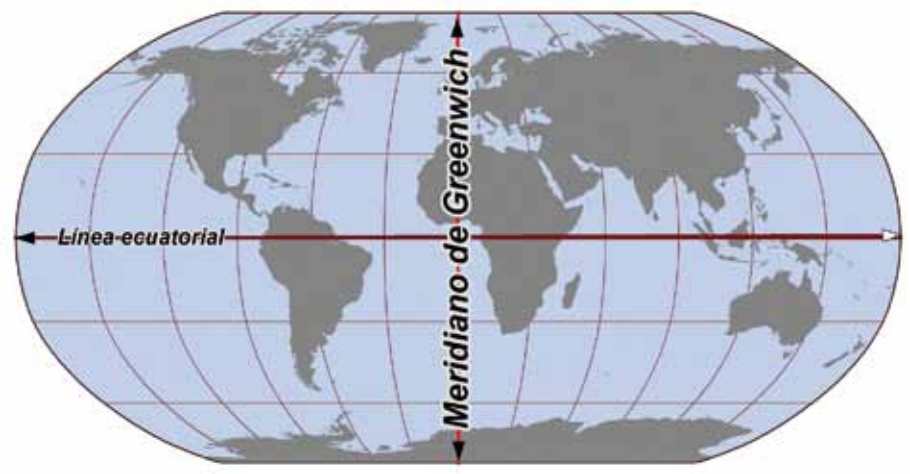
De forma prática, um servidor NTP fica escutando na porta 123/UDP aguardando solicitações de clientes. Quando um cliente aciona o cliente do sistema operacional em busca de atualização em um *stratum* superior ao seu, inicia-se uma transação buscando fornecer ao cliente uma hora exata, usando-se de algoritmos de compensação de latência de rede, o servidor de tempo entrega a hora para o sistema operacional cliente.

Mas será que existe um padrão de tempo? No que o horário é baseado? É isso que você verá agora.

13.3 PADRÕES DE TEMPO

O horário no mundo é baseado em dois padrões de tempo: o Tempo Médio de Greenwith – *Greenwich Mean Time* (GMT), ou no Tempo Universal Coordenado – *Universal Time Cordinated* (UTC). No primeiro deles, os horários são definidos para leste ou oeste do meridiano, sendo que para leste é positivo e, para oeste, negativo.

Confira a figura ilustrativa do GMT.



fotógrafo

Figura 35 - Meridiano de Greenwich

O meridiano que *Greenwich* é uma marca "imaginária" que cruza a cidade de Greenwich, perto de Londres, na Inglaterra. Este meridiano divide o mundo em ocidente e oriente, permitindo, dessa forma, mensurar a longitude de pontos. Ele foi definido como padrão e serve como marco oficial para determinar o fuso horário dos países no mundo.

Na figura a seguir, você pode ver o observatório de *Greenwich*, onde a linha "imaginária" cruza.

fotógrafo

Figura 36 - Meridiano 0, marcado no Observatório de *Greenwich* ao Leste de Londres

Na virada do ano de 1971 para 1972, o antigo GMT foi trocado por um novo padrão que viria a ser adotado como referência de tempo, o UTC. O padrão de tempo UTC é mantido pelo Bureau Internacional de Pesos e Medidas. Ele ainda é padrão para muitos sistemas e protocolos na Internet, em particular, e o *Network Time Protocol* (NTP) foi desenvolvido tendo o UTC como padrão.

CASOS E RELATOS

O descuido que saiu caro

Uma empresa da área de alimentos, que tem um grande setor na área de pesquisa onde são concebidos novos produtos para a mesma, sempre foi vítima de tentativas de espionagem industrial por empresas concorrentes e sempre sofria com as constantes tentativas de ataque aos sistemas computacionais. Para tentar minimizar o problema, investiu em infraestrutura comprando novos servidores e sistemas de armazenamento de dados, a fim de diminuir o tempo de parada caso ocorresse alguma falha de segurança. No entanto, certo dia um invasor encontrou uma falha no servidor *web* e acessou o *Shell* da empresa. A partir daí, ele foi escalando, servidor por servidor, buscando falhas de segurança em cada um deles, até que conseguiu chegar ao servidor de produção que dava suporte às pesquisas e furtou todo o banco de dados. A equipe que administrava a rede não tinha experiência suficiente para analisar o ataque e detectar o que havia acontecido, então, contrataram uma empresa terceirizada de segurança de redes. Os especialistas perceberam que cada servidor tinha uma hora diferente e isso inviabilizaria a checagem de onde o invasor passou e os tempos de permanência em cada *host*, para, dessa forma, traçar o seu rastro. Surpresos pela empresa ter gasto milhões em infraestrutura, mas não ter um servidor NTP para sincronizar os sistemas computacionais, informaram à empresa que esse pequeno detalhe se tornou uma falha grave na área de gerência de servidores de redes, porque é necessário manter os relógios das máquinas atualizados. A partir daquele momento, o administrador da rede passou a cuidar desse importante componente de rede e, com a implementação do serviço, a sincronização dos relógios foi feita em poucas horas.

13.4 *SOFTWARES* PARA SERVIDORES DE TEMPO

Para sistemas UNIX em geral, tais como FreeBSD, Linux, Mac OS X, AIX, HP-UX, nós temos a disposição o *ntpd* ou NTP Server que faz a função de servidor de tempo NTPv4 para clientes UNIX e Windows. Se utilizarmos Windows como plataforma, a partir do Windows 2000 Server já vem com a aplicação Serviço de Tempo do Windows, que pode ser usado para o mesmo fim. Este último foi bastante criticado, no início, pois não seguia a versão sugerida na RFC 1305 para o NTPv3. A partir do Windows 2003 Server, o Serviço de Tempo do Windows vem, em sua documentação, que segue a versão sugerida nas RFCs.

FIQUE ALERTA

Se um servidor *Stratum* 1 não estiver atualizado, ele não fornecerá tempo por razões de segurança. Fique alerta, caso seu servidor não esteja atualizando os clientes, pois esse pode ser o motivo.

13.5 INSTALAÇÃO DE UM SERVIDOR DE TEMPO

Para exemplificar a instalação de um servidor de tempo, utilizaremos um que se baseia no protocolo NTYP, chamado NTP Server, ou "ntpd". Como sistema operacional hospedeiro, utilizaremos um Debian Linux/GNU versão *Squeeze*. Confira a instalação, a seguir.

```
root@server:/# apt-get install ntp
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
Os pacotes extra a seguir serão instalados:
  libopts25
Pacotes sugeridos:
  ntp-doc
Os NOVOS pacotes a seguir serão instalados:
  libopts25 ntp
0 pacotes atualizados, 2 pacotes novos instalados, 0 a
serem removidos e 2 não atualizados.
É preciso baixar 566 kB de arquivos.
Depois desta operação, 1303 kB adicionais de espaço em
disco serão usados.
```

```
   Você quer continuar [S/n]? S
   Obter:1  http://debian.pop-sc.rnp.br/debian/  squeeze/
main libopts25 amd64 1:5.10-1.1 [66,6 kB]
   Obter:2  http://debian.pop-sc.rnp.br/debian/  squeeze/
main ntp amd64 1:4.2.6.p2+dfsg-1+b1 [499 kB]
   Baixados 566 kB em 0s (3709 kB/s)
   Selecionando  pacote  previamente  não  selecionado
libopts25.
   (Lendo banco de dados ... 25336 ficheiros e directórios
actualmente instalados.)
   Desempacotando libopts25 (de .../libopts25_1%3a5.10-1.1_
amd64.deb) ...
   Selecionando pacote previamente não selecionado ntp.
   Desempacotando ntp (de .../ntp_1%3a4.2.6.p2+dfsg-1+b1_
amd64.deb) ...
   Processando gatilhos para man-db ...
   Configurando libopts25 (1:5.10-1.1) ...
   Configurando ntp (1:4.2.6.p2+dfsg-1+b1) ...
   Starting NTP server: ntpd.
```

Agora, já temos o servidor de tempo "ntpd" rodando em nosso sistema. Por padrão ele já vem configurado para ser funcional desde o início. Toda configuração do servidor de tempo "ntpd" no Linux está localizada no arquivo "ntp.conf", localizado no diretório "/etc". Alguns parâmetros importantes que podem ser modificados ou inseridos, neste arquivo de configuração, podem ser vistos no quadro a seguir.

PARÂMETRO	DESCRIÇÃO
server	Este parâmetro determina em que servidor *stratum* o nosso servidor irá buscar atualizações. Nós podemos inserir várias linhas com endereços de servidores de tempo, pois caso haja uma falha, ele buscará no próximo da lista.
broadcast	O servidor de tempo fará broadcasting para a rede local.
disable auth	Desabilita autenticação sugerida para atualização de tempo.
restrict	Aplica restrições de acesso ao servidor de tempo, por exemplo: *restrict default ignore*. Desta forma, o servidor irá negar atualizações para todos. Este parâmetro é bastante utilizado para restrições de redes e *hosts* que podem acessar o servidor.

Quadro 15 - Algumas configurações do "ntp.conf"

SAIBA MAIS

Existem muitas outras configurações que podem ser realizadas no arquivo de configuração do servidor de tempo. Faça uma visita ao *site* <www.ntp.org> e confira.

Depois de feita a configuração, é preciso aguardar alguns instantes para que o nosso servidor se atualize nos servidores (server) que listamos no arquivo de configuração. Após esta etapa, o nosso servidor NTP estará pronto para fornecer atualizações de tempo para os clientes.

Veja um exemplo de solicitação de atualização no Linux:

```
root@server:/# ntpdate ntp.exemplo.com
16 Sep 14:06:14 ntpdate[10045]: adjust time server
200.144.121.33 offset -0.007469 sec
```

RECAPITULANDO

Neste capítulo, você conheceu aspectos relacionados a servidores de tempo ou servidores de sincronismo de relógios(que são sinônimos), e viu que eles usam como base o protocolo NTP, que atualmente está em sua versão 4 (NTPv4). Conheceu um pouco do histórico deste tipo de servidor, seu modo de operação, baseado em uma hierarquia de servidores de tempo (stratum) que, por sua vez, usam como base os servidores de referência: relógios atômicos, GPS ou receptores. Ainda neste capítulo, você conheceu os padrões de tempo, que antigamente usavam o GMT, mas hoje o padrão de Internet utilizado é o UTC, e que o próprio NTP Server é implementado em cima desse padrão. Por fim, você conheceu algumas ferramentas para implantar servidores de tempo e visualizou uma instalação básica usando o "ntpd" sobre o sistema operacional Debian Linux/GNU na versão Squeeze. Agora, respire fundo e inicie mais uma jornada, conhecendo os servidores de Logs. Até mais!

Servidor de Logs (Syslog)

14

Neste capítulo, você conhecerá um pouco da história do Syslog, os motivos que culminaram em seu surgimento, o seu modelo de operação, abordando as facilidades e os níveis de prioridade e, por fim, verá um exemplo sobre as regras que podem ser definidas no arquivo de configuração do *syslog*. Ao final deste capítulo, você terá subsídios para:

a) conhecer a evolução e a história dos servidores de Log;

b) entender o modelo de operação do Syslog;

c) compreender a utilização do Syslog.

É importante que você entenda bem este tipo de servidor pois esse conhecimento será crucial para proporcionar um maior nível de controle sobre os serviços da rede. Portanto, muita atenção e bom estudo!

14.1 HISTÓRIA

Em um sistema operacional, a cada segundo, ocorre uma série de eventos e atividades que, na maioria das vezes, nós nem percebemos. Essas atividades são chamadas de logs e são registradas por um serviço do sistema operacional.

Os logs do sistema são muito importantes sob vários aspectos, desde a auditoria do sistema até as análises forenses que venham a acontecer. Desta forma, é de extrema importância que o sistema operacional esteja com o serviço de *logs* ativo e configurado para monitorar o sistema.

Para sistemas operacionais Unix-like, como Debian, Red Hat, AIX, HP-UX, Mac OS X, entre outros, o serviço mais conhecido e utilizado para servidores de log é o *syslog*. Para a plataforma Windows, este sistema operacional já vem em suas versões Server com o Serviço de Eventos do Windows como padrão, que tem a responsabilidade de registrar os *logs* do sistema operacional e dos serviços a ele plugados.

O *syslog* foi desenvolvido em 1980, por Eric Allman, como parte do projeto Sendmail (sistema de *e-mail*) e, na época, era usando somente por este sistema de *e-mail*. Como demonstrou ser muito flexível e outras aplicações começaram a utilizá-lo como sistema de registros de eventos-padrão, acabou se tornando o sistema de registro-padrão para sistemas baseados em Unix. Lembre-se de que existe uma grande variedade de implementação do syslog para outros sistemas operacionais. Assim sendo, a IETF (*The Internet Engineering Task Force*) definiu o padrão na RFC 3164 e, desde então, várias extensões e melhorias foram propostas para o padrão, modificando assim a RFC 3164 e tornando-a obsoleta, desde o lançamento da RFC 5424.

É importante que você saiba que muitas empresas adotam o sistema de registros de *log syslog* dentro de seus equipamentos, tais como roteadores, *switchs* e afins. No item a seguir, conheça o modelo do *Syslog*.

VOCÊ SABIA?

Você sabia que é possível extrair a data e hora exata de cada evento no sistema operacional? Isto é possível utilizando os logs armazenados no /var/log.

14.2 MODELO DO SYSLOG

Na operação do *syslog*, as mensagens de eventos são referenciadas por "facilidades" (*facility*), que podem ser: auth, authpriv, daemon, cron, ftp, lpr, kern, mail, news, syslog, user, uucp, locals, e são definidos níveis de prioridade para cada uma destas facilidades. Mais adiante, será exemplificada a utilização das facilidades e onde elas se encaixam no cenário.

No quadro a seguir, é possível visualizar a lista de níveis e suas definições.

NÍVEL	DESCRIÇÃO
Debug (7)	Informações sobre depuração (debug).
Info (6)	Mensagens de informação.
Notice (5)	Mensagens sobre condições normais de operação, mas com certa importância.
Warning (4)	Condições de alerta.
Err (3)	Condições de erros.
Crit (2)	Condições de operação crítica.
Alert (1)	Ações imediatas são requeridas.
Emerg (0)	Sistema indisponível.

Quadro 16 - Níveis do *Syslog*

O *syslog* pode trabalhar como servidor centralizador de logs, desta forma ele recebe todos os logs de máquinas remotas e armazena os logs. O armazenamento pode ser feito no próprio sistema de arquivos ou ainda utilizar sistemas gerenciadores de bancos de dados para esta função. A implementação do *syslog* que provê suporte a banco de dados é chamado *Remote Syslog* ou simplesmente *rsyslogd*.

Confira! Na figura a seguir, você pode ver um exemplo da utilização de um centralizador de *logs*, utilizando *rsyslog* ou alguma outra implementação do *syslog* compatível com suporte a banco de dados.

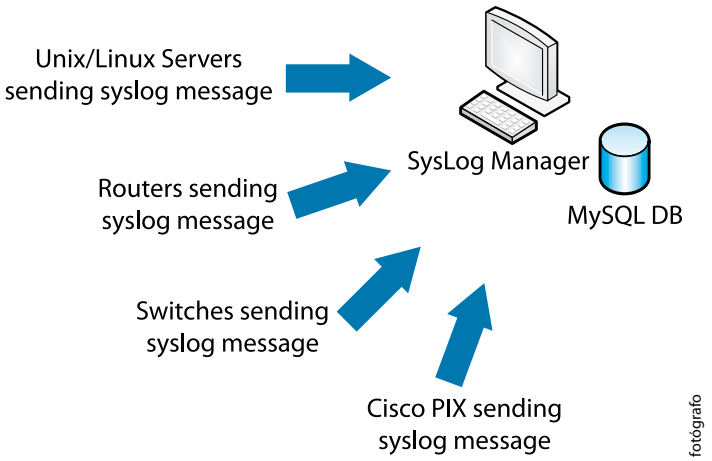


Figura 37 - Exemplo de Arquitetura Centralizada

Nessa figura, servidores de redes, roteadores, *switches* e *firewalls* estão enviando os registros dos eventos para o centralizados de *logs*, que está armazenando em um banco de dados MySQL.

FIQUE ALERTA

Os sistemas de logs são de extrema importância no âmbito dos servidores de redes. Em algum momento você, com certeza, irá necessitar das informações dos logs, então, esteja sempre atento com este componente!!

14.3 EXEMPLO DE SYSLOG

Como você já viu, o *syslog* tem várias implementações que diferem em suporte a banco de dados e diferentes sistemas operacionais até o suporte a extensões para outros tipos de dados. É importante salientar que todo sistema operacional de rede por padrão deve ter um sistema gerenciador de *logs*. No caso da versão *Squeeze* do Debian GNU/Linuxele já vem instalado na versão *rsyslogd*, para dar suporte a *logs* remotos.

Para complementar o estudo sobre *syslog*, acompanhe o exemplo do Casos e relatos a seguir.

CASOS E RELATOS

Integração de registros para segurança

Em uma cooperativa de agropecuária, o administrador de rede tinha o desafio de integrar múltiplos servidores de suas filiadas, buscando um registro exato de cada evento ocorrido em cada delas. Isso serviria para fatores de futuras auditorias, caso viesse a ocorrer alguma falha de segurança ou, ainda, se precisassem analisar alguma situação estranha que surgisse. O problema estava em como integrar esses registros de *log*. Como todas as filiadas trabalhavam com servidores de rede Linux, o administrador de redes foi taxativo e indicou a utilização do *syslog* para integração de todos os registros. Assim, ele implantou um servidor de *logs* dedicado somente para integração de todos os registros na sede da cooperativa. Dessa forma, cada novo evento que ocorresse no sistema operacional e nas aplicações seriam direcionados para o servidor de *logs* da sede. Essa solução resolveu o problema de registros e atendeu todos os requisitos exigidos.

Na figura a seguir, você pode ver um exemplo de arquivo de configuração do *syslog*, onde estão sendo apresentadas as regras que definirão o que será registrado, a prioridade e o local onde o *log* será armazenado. Veja que todos os logs estão sendo direcionados para a pasta "/var/log", que, por padrão, é a pasta onde os *logs* devem ser armazenados em sistemas operacionais Linux.

```
###############
#### REGRAS ###
###############

#
# First some standard log files.  Log by facility.
#
auth,authpriv.*              /var/log/auth.log
*.*;auth,authpriv.none            -/var/log/syslog
#cron.*                      /var/log/cron.log
daemon.*                -/var/log/daemon.log
```

```
kern.*                          -/var/log/kern.log
lpr.*                           -/var/log/lpr.log
mail.*                          -/var/log/mail.log
user.*                          -/var/log/user.log

#
# Logging for the mail system.  Split it up so that
# it is easy to write scripts to parse these files.
#
mail.info               -/var/log/mail.info
mail.warn               -/var/log/mail.warn
mail.err                /var/log/mail.err

#
# Logging for INN news system.
#
news.crit               /var/log/news/news.crit
news.err                /var/log/news/news.err
news.notice                     -/var/log/news/news.notice

#
# Some "catch-all" log files.
#
*.=debug;\
    auth,authpriv.none;\
    news.none;mail.none     -/var/log/debug
*.=info;*.=notice;*.=warn;\
    auth,authpriv.none;\
    cron,daemon.none;\
    mail,news.none          -/var/log/messages
```

Viu só como é simples a implementação do *syslog*? Ele é definido, basicamente, no formato:

Facilidade > Prioridade > Arquivo onde será armazenado

Esta característica torna o *syslog* flexível e de fácil utilização, visto que é possível definir uma facilidade e definir onde e como se quer armazenar cada uma das prioridades para uma melhor gerência ou, ainda, usar a expressão regular asterisco (*) para definir que você deseja que todas as prioridades sejam armazenadas no mesmo arquivo de *log*.

SAIBA MAIS

O *syslog* é um importante componente da arquitetura de redes de computadores e, para você saber mais informações importantes sobre ele, acesse o *site* <http://www.syslog.org>.

RECAPITULANDO

Neste capítulo, você acompanhou uma introdução sobre a importância dos sistemas de registro de eventos nas infraestruturas das redes de computadores, chamados *logs*. Conferiu a história do *syslog*, que surgiu como componente base do sistema de *e-mail Sendmail* e que, rapidamente, se tornou padrão nos sistemas operacionais. Conheceu um modelo de operação do *syslog*, que funciona com um esquema de facilidade (*facility*) e prioridades determinadas para cada uma das facilidades que são armazenadas em arquivos. E, para finalizar, viu um exemplo sobre regras do syslog, baseados no sistema operacional Debian GNU/Linux na versão *Squeeze*. No próximo capítulo, você conhecerá um serviço de gerenciamento de pacotes. Continue atento e bom estudo!

Serviço de Atualização de Patchs

15

Neste capítulo, você conhecerá o serviço de gerenciamento de pacotes de sistemas operacionais *Debian-like* (sistemas baseados na distribuição Debian), chamado APT. Verá, também, o serviço de gerenciamento de pacotes YUM para sistemas operacionais *Red Hat-like* (sistemas operacionais baseados em *Red Hat*) e, por fim, o *Windows Update*, que é o serviço de atualização para sistemas operacionais *Windows-like*.

Ao final desse capítulo, você terá subsídios para:

a) compreender os aspectos dos servidores de atualização de *patches*;

b) entender o funcionamento do APT;

c) conhecer o funcionamento do YUM;

d) entender o funcionamento do Windows Update.

Esse tipo de serviço é muito importante para bons níveis de segurança nos serviços de redes da organização. Portanto, procure manter-se bem atento na leitura dos conteúdos apresentados.

15.1 DEFINIÇÕES

A área de sistemas operacionais de rede é crítica no que diz respeito à segurança. Quando decidimos disponibilizar um serviço via rede, temos que nos preocupar com os aspectos relacionados à segurança dos serviços. Isto inclui a utilização de mecanismos de segurança como IDS (*Intrusion Detection Systems*), IPS (*Intrusion Prevent Systems*), *Firewalls*, entre outros. Entretanto, há um quesito que é primordial mas alguns administradores de rede não dão a importância necessária: a atualização dos sistemas.Entende-se por atualização de *paths* o método de atualização de pacotes dos sistemas operacionais. Um *path*, neste caso, nada mais é do que uma correção/extensão de um *software*. Essas atualizações geralmente não tem uma periodicidade para acontecer, salvo alguns casos de atualizações programadas, e servem para atualizar:

a) funcionalidades já existentes;

b) novas funcionalidades (melhorias);

c) correção de funcionalidades;

d) correção de brechas de segurança;

Então, quando você tem um serviço de um *software*, é importante mantê-lo atualizado para o seu perfeito funcionamento, pois, não adianta termos os melhores mecanismos de segurança existentes, se o *software* que está fornecendo o serviço apresenta um problema de segurança que pode abrir uma brecha para uma invasão ao sistema e comprometer alguns ou todos os serviços.

Com a evolução da Internet, os distribuidores de sistemas operacionais como Debian, Red Hat, Microsoft, entre outros, verificaram a necessidade de incluir em seus sistemas operacionais um serviço que possibilitasse fazer todas estas atualizações de melhorias, *bugs* e segurança via rede, de uma forma segura, possibilitando ao usuário que fizesse as atualizações necessárias.

Surgiram, então, algumas ferramentas de atualização, como o APT, YUM e Windows Update, que você conhecerá mais detalhadamente nas etapas a seguir.

15.2 APT

A Ferramenta Avançada de Empacotamento ou *Advanced Packaging Tool* (APT) é uma interface com o usuário que tem a função de gerenciar a instalação, atualização e remoção de *software* em sistemas operacionais baseados em Debian Linux/GNU. O APT simplifica o processo de gerência de *softwares* para o sistema operacional baseando-se num esquema de recuperação de pacotes de repositórios remotos, checagem de dependências e manutenção de uma base de pacotes instalados.

O APT foi concebido incialmente para ser integrado com o "dpkg" (*Debian Package Management System*), que é o sistema que gerencia os pacotes, localmente, em sistemas operacionais do tipo Debian. Porém, com o tempo, ele foi estendido para trabalhar com o gerenciador de pacotes para distribuições baseadas em Red Hat, chamado "rpm" (RedHat Package Manager) o que originou o "apt-rpm", que é uma extensão para trabalhar com o formato RPM.

Todo sistema baseado em Debian já vem com o APT instalado em seu sistema. Os arquivos e diretórios que compõem a configuração da ferramenta APT ficam localizados no diretório "/etc/apt/", ou no diretório "/var/lib". A lista de arquivos e diretórios pode ser vista no quadro a seguir.

ARQUIVOS E DIRETÓRIOS	DESCRIÇÃO
/etc/apt/sources.list	Arquivo onde são configurados os locais onde o APT pegará as atualizações.
/etc/apt/sources.list.d/	Diretório onde podem ser colocadas informações adicionais do "source.list".
/etc/apt/apt.conf	Arquivo principal de configuração do APT.
/etc/apt/apt.conf.d/	Diretório onde podem ser colocadas informações adicionais do "apt.conf".
/var/cache/apt/archives/	Área de armazenamento dos arquivos que foram baixados pelo APT.
/var/cache/apt/archives/partial/	Área de armazenamento de pacotes que ainda não foram baixado completamente.

Quadro 17 - Arquivos de Configuração do APT

O APT funciona em um esquema de comandos e ações. Quando determinado comando, é dada a ferramenta que executa a ação desejada pelo usuário. Estes comandos cobrem a parte de instalação, remoção e atualização de pacotes do sistema operacional. Ainda é possível executar ações que afetem todo o sistema operacional, como uma atualização completa. Veja, a seguir, alguns exemplos de parâmetros possíveis:

e) Install: esse parâmetro é utilizado para instalação e atualização de pacotes. Quando executado sobre um pacote já existente, ele irá checar se a versão do pacote local é a mesma do pacote que há nos servidores de atualização listados no source.list. Se a versão for inferior, ele irá sugerir a atualização para a versão mais atual. Se efetuado em um pacote que não esteja instalado no sistema operacional, ele executará a instalação das dependências e dos pacotes em si relacionados. Se executado sem o *purge*, o APT manterá os arquivos de configuração. Exemplo: **# apt-get install bind9**.

f) Remove: é utilizado para remover pacotes que estejam instalados no sistema operacional. Por padrão, ele não remove arquivos de configuração, somente o programa em si, mas é possível remover os arquivos de configuração também com um parâmetro auxiliar ao comando remove, o *purge*. Por exemplo: **# apt-get --purge remove bind9**.

g) Update: o parâmetro update é utilizado para atualizar a base de pacotes do APT. O APT mantém uma base de dados local dos pacotes que os servidores listados no source.list têm disponíveis para operação. Uma vez executado o *update*, esta base será sincronizada com a base do servidor. Exemplo: **# apt--get update**.

h) Upgrade: esse parâmetro é utilizado para atualização do sistema. Uma vez executado, o APT irá recuperar a lista de pacotes e versões do sistema operacional e cruzar com a lista de pacotes e as versões do servidor de atualização, atualizando, dessa forma, todos os pacotes do sistema que estejam dentro da mesma versão de sistema operacional. Exemplo: **# apt-get upgrade**.

Dist-upgrade: utilizado para atualização de versões do sistema operacional. Da mesma forma que o *upgrade*, ele faz uma listagem de pacotes e versões do sistema e recupera a listagem dos mesmos pacotes da versão que está listada no "source.list". É utilizado para atualização do sistema operacional como um todo. Exemplo: **# apt-get dist-upgrade**

SAIBA MAIS

Mais informações sobre o APT podem ser encontradas no *site* oficial do sistema operacional Debian Linux/GNU: <www.debian.org>.

O APT ainda tem outros comandos relacionados à sua operação, como por exemplo, a gerência de *cache* (*apt-cache*), sincronia a partir do CD-ROM (*apt--cdrom*), gerência de chaves de repositórios (*apt-key*), configuração do APT (*apt--config*), entre outros. O importante é que você saiba identificar que é possível fazer atualizações *patches* a partir deste importante serviço.

Acompanhe um exemplo de atualização de patche no Casos e relatos a seguir, e veja o quanto as atualizações são importantes.

CASOS E RELATOS

Atenção na atualização de patches

No parque computacional de uma universidade havia um servidor de *e-mails* que era específico para os colaboradores da instituição. Como os professores utilizavam o serviço, esse servidor era alvo de constantes ataques dentro da rede. Suspeitava-se que os ataques eram feitos por alunos que buscavam informações úteis para seus estudos, ou mesmo para fins destrutivos, e, por isso, o administrador da rede sempre tinha um cuidado redobrado com este servidor. Em uma sexta-feira, à noite, quando o administrador de redes foi para casa tranquilamente, um grupo de *crackers* da Internet descobriu uma falha de segurança no *software* de gerência de e-mails que permitia a criação de um *exploit* que dava acesso à *Shell* do servidor, com acesso de administrador. A comunidade de segurança de redes ficou sabendo da notícia e tratou de espalhá-la rapidamente na Internet. Em duas horas, a comunidade de desenvolvedores disponibilizou um *patch* de segurança que corrigia o problema do *software*, atualizando o servidor que, assim, salvou-se da ameaça. Porém, o administrador de rede não ficou atento a esse tipo de ameaça e, na segunda-feira pela manhã, quando chegou na instituição, ao checar seus *e-mails* percebeu que havia vários alertas dos órgãos de segurança da Internet para a falha que havia ocorrido. Imediatamente ele atualizou o sistema de *e-mails* e iniciou uma varredura nos registros (*logs*) das transações do final de semana, constatando que a invasão havia ocorrido no sábado, e os dados dos professores e funcionários haviam sido furtados. Provas, trabalhos, dados administrativos, planejamento de ações, enfim, todas as informações contidas nos *e-mails* institucionais foram comprometidas. A direção foi imediatamente comunicada, e, posteriormente, todos os envolvidos foram informados sobre o ocorrido, causando um grande transtorno. Depois desse episódio, o administrador passou a ficar muito mais atento aos canais de segurança disponíveis para evitar que esse problema voltasse a acontecer.

Além da ferramenta APT, que você acabou de conhecer, falaremos sobre outras duas neste material. A próxima é a YUM – *Yellow dog Updater Modified*. Quer saber o que ela faz? Confira no item a seguir.

15.3 YUM

A *Yellow dog Updater Modified,* ou simplesmente YUM, é uma ferramenta utilizada para instalação, remoção e atualização de pacotes para sistemas operacionais do tipo *Red Hat-like*, como o *Red Hat Enterprise*, *CentOS*, *Fedora*, entre outros. Ela é baseada em comandos e ações, e suporta parâmetros para customização das ações. Seus arquivos de configuração mudam de distribuição para distribuição, mas no geral, seus arquivos são os listados no quadro a seguir.

ARQUIVOS E DIRETÓRIOS	DESCRIÇÃO
/etc/yum	Diretório que contém arquivos de plugins ou configuração.
/etc/yum.conf	Arquivo de configuração principal do YUM
/etc/yum.repos.d/	Diretório que contém os arquivos de configuração dos servidores de atualização.

Quadro 18 - Arquivos e diretórios do YUM

Baseadas nas configurações realizadas, o YUM pode, agora, executar operações no sistema operacional. Essas operações cuidam da instalação, remoção e atualização de pacotes para o sistema operacional. Confira, a seguir, uma lista de parâmetros.

a) **Install**: esse parâmetro é utilizado para instalar pacotes no sistema operacional. O modo de operação é checar a existência do pacote no repositório remoto configurado, fazer o *download* do pacote e suas dependências e instalá-lo. Exemplo: **# yum install bind**.

b) **Remove**: o parâmetro remove faz a remoção (desinstalação) de pacotes. É importante salientar que ele não remove as dependências dos pacotes, fato que, com o tempo, pode ocasionar inconsistências no sistema de pacotes. Exemplo: **# yum remove bind**.

c) **Update**: este parâmetro atualiza a lista de pacotes instalados no sistema operacional. Uma vez executado o YUM, chegarão todos os pacotes do sistema e vão comparar suas versões com a lista de pacotes e versões dos repositórios. Exemplo: **# yum update**.

d) **Upgrade**: este parâmetro atualiza a lista de versões de pacotes do repositório remoto com a base de dados local. Exemplo: **# yum upgrade**.

O sistema de gerenciamento de pacotes YUM ainda é muito utilizado. Ele solidifica o processo de gerência de servidores, proporcionando facilidades aos administradores de rede que não precisam se preocupar em atualizar manualmente, pacote por pacote do sistema operacional. Como observação, muitos administradores não utilizam o YUM por acharem ele menos flexível do que o APT, desta forma, mesmo em sistemas operacionais baseados em *Red Hat*, é comum usarmos o "apt-rpm" para gerência de pacotes.

E, para quem utiliza o sistema operacional Windows, a ferramenta de atualização é o Windows Update. Veja mais informações sobre ele no próximo item.

15.4 WINDOWS UPDATE

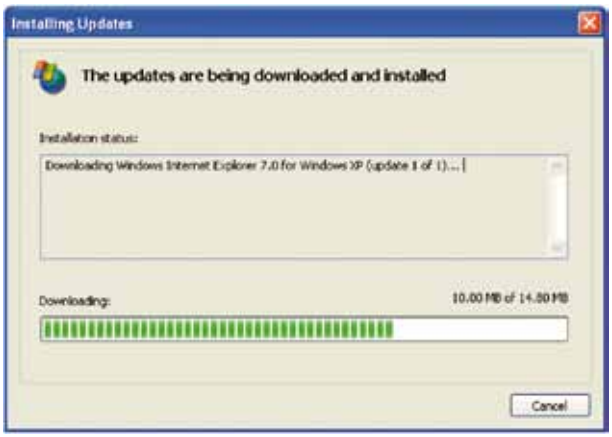
O Windows Update é o serviço que provê atualizações para sistemas operacionais da família Windows, tais como: Windows XP, Windows Vista, Windows 2008 Server, entre outros. O serviço de atualização Windows Update facilita o processo de atualização do sistema operacional Windows. Diferentemente dos sistemas gerenciadores de pacotes para sistemas Linux, o Windows Update só se preocupa com os *softwares* do próprio sistema operacional, com exceção do Microsoft Office.

FIQUE ALERTA

Quando você estiver executando servidores de redes baseados em Windows, é muito importante se preocupar com os mecanismos de segurança e ficar atento às atualizações da Microsoft para evitar possíveis ataques.

Quando a Microsoft detecta algum problema de segurança ou funcionalidade do sistema operacional, ela disponibiliza esses *patches* em seus repositórios para que os usuários atualizem seus sistemas operacionais. É possível deixar este processo automático, desta forma, na maioria dos casos o usuário nem percebe que o seu sistema operacional está baixando as atualizações e instalando-as.

Por padrão, assim que um Windows é instalado, o sistema de instalação questiona que tipo de modelo de atualização quer usar, podendo ser: automático, semi-automático ou desativado. Em linhas gerais estes modelos podem mudar de Windows para Windows. Na figura a seguir é possível ver o Windows Update rodando em um sistema operacinal Windows XP.



fotógrafo

Figura 38 - Exemplo do *Windows Update*

Uma crítica forte da comunidade de *software* livre para com este serviço é que como os Windows são sistemas operacionais fechados, o seu desenvolvimento é controlado, assim como suas atualizações. Desta forma, a única maneira de se ter o sistema consistente e atualizado é utilizando esta ferramenta. Há casos – principalmente quando a Microsoft divulga os conhecidos *Service Packs* (pacotes de serviços), que são pacotes com várias atualizações dentro dele – em que o usuário pode baixar e instalar os pacotes sem o auxílio do *Windows Update*.

Seguindo a crítica, quando se trata de servidores de rede, a utilização de sistemas baseados em *Windows* deve ser cautelosa, porque já houve casos de falhas de segurança serem descobertas pela comunidade, e estas falhas demorarem duas ou três semanas para serem fixadas pelo *Windows Update*. Desta forma, os administradores ficaram quase três semanas com os servidores de rede expostos a possíveis ataques.

VOCÊ SABIA?

Você sabia que a Microsoft tem outro serviço de atualização, chamado *Windows Server Update Services* (WSUS)? Apesar de ele usar a mesma base de conhecimento da Microsoft é destinado ao mercado corporativo de servidores de rede.

O *Windows Update* é uma boa ferramenta para atualização de sistemas operacionais. Ela foi e é de grande valia para auxiliar usuários que não têm experiência, aqueles usuários que não querem se preocupar com quesitos de atualizações do sistema operacional. Desta forma o *Windows Update* se destaca como ferramenta de atualização para sistemas Windows, sendo ela a única opção para este tipo de operação.

RECAPITULANDO

Neste capítulo, você viu alguns aspectos relacionados a serviços de atualização de patches para sistemas operacionais e pacotes. Conheceu o *Advanced Package Tool*, o APT, que é o sistema de gerenciamento de pacotes para sistemas baseados no sistema operacional *Debian*. Ainda, viu o *Yellow dog Updater*, *Modified*, o YUM, que é um sistema gerenciador de pacotes usado para sistemas operacionais baseados em *Red Hat*, e o *Windows Update*, que gerencia atualizações de sistema operacionais da família Windows e proporciona aos usuários atualizações de melhorias e segurança de modo fácil e simples.

Agora que você já conheceu todas essas informações importantes sobre os serviços de redes, que tal conhecer um pouco sobre *backup*? Esse é outro assunto muito importante, afinal, é sempre bom garantir que seus arquivos fiquem bem armazenados, não é mesmo? Então, siga em frente!

Mecanismos de *Backup*

16

Neste capítulo, você conhecerá aspectos relacionados a mecanismos de *backup*, iniciando pelos tipos mais conhecidos, os meios de armazenamentos mais comuns, passaremos por uma seção de exemplos de ferramentas para implantação de *backup* para uma grande variedade de sistemas operacionais e, por fim, verá um exemplo de instalação para transferência e sincronia de arquivos a serem *backupeados* usando o "rsync".

Ao final deste capítulo, você terá subsídios para:

a) conhecer os aspectos relacionados a mecanismos de *backup*;

b) entender os tipos de *backup*;

c) conhecer os tipos de meios de armazenamento de *backups*; e

d) compreender os mecanismos de *backup*.

A compreensão destes tipos de mecanismos é fundamental para que o administrador possa implementar boas rotinas de *backup* na organização, preservando assim o principal ativo nas organizações de hoje em dia: as informações. Preparado para começar este último capítulo? Então, vamos lá!

16.1 TIPOS DE *BACKUPS*

Uma cópia de segurança ou, como é comumente chamado "processo de *backup*", é o ato de realizar cópias de dados que podem ser usados para restaurar o estado original do sistema, após algum tipo de evento que venha a comprometer os dados, como por exemplo, a queima de discos rígidos, ataques que comprometam os bancos de dados ou qualquer outro evento a que estamos sujeitos, a partir do momento que disponibilizamos serviços na Internet.

Copiar dados não é uma tarefa muito fascinante, mas é uma tarefa importante para a confiabilidade, a longo prazo, de seus sistemas. Algumas redes consistem em um ou dois grandes servidores e uma grande quantidade de sistemas menores dentro destes servidores. Dessa forma, a tarefa de *backup* é essencial para a segurança dos dados.

Entretanto, quando aumentamos o tamanho da rede em questão, as rotinas de *backup* são ainda mais importantes, visto que a perda de dados dentro de grandes organizações é algo impensável. Assim, é possível aplicar sólidas rotinas de *backup*, com soluções livres ou pagas, garantir que os dados da organização estejam preservados.

Estratégias de *backup* geralmente devem iniciar com um modelo de repositório de dados onde os administradores irão definir onde e como os esquemas de backup serão realizados. Os dados que compõem o *backup* devem ser guardados e organizados seguindo uma estrutura e um local pré-definidos.

Nesse material, falaremos sobre os seguintes tipos de *backup*: não-estruturados, *full backup*, incremental e diferencial. Antes de conhecer melhor cada um deles, acompanhe um Casos e relatos para compreender melhor a sua importância.

CASOS E RELATOS

14% de perda, por descuido

Uma empresa da área têxtil, da cidade de Blumenau, teve um sério problema nas rotinas de *backup* de seus servidores. A empresa tinha mais de 40 anos de história e teve a preocupação de passar todo o seu histórico, desde dados de pessoas que passaram pela empresa (RH) até o histórico financeiro e administrativo, para o meio digital.

O processo antigo era realizado manualmente por seis administradores de rede, mas há um ano atrás, a empresa adotou uma solução proprietária que automatizava todo o processo de *backup* de seus bancos de dados. O problema é que, com o tempo, os administradores passaram a confiar demasiadamente na solução e pararam de realizar as rotinas de verificação de *backup* semanais que estavam planejadas. Certo dia, de madrugada, o administrador sênior recebeu uma mensagem de texto no seu celular, vinda diretamente do sistema de monitoramento da rede com a seguinte informação: "Servidor de banco de dados *off-line*". Como a empresa trabalha 24 horas por dia, durante os sete dias da semana, 365 dias por ano, o administrador pulou da cama e foi até o seu computador para tentar acessar o servidor, mas não conseguiu, pois o mesmo estava comprometido. Chegando à empresa, uma hora depois, a primeira análise foi de que os discos de dados estavam queimados. Foi então acionado o servidor que já estava preparado para contingenciar o servidor de banco de dados principal. Quando estava tudo pronto para começar o processo de restauração do *backup*, apareceu a seguinte mensagem: "*Backup corrompido*". Obviamente havia uma cópia semanal do backup, mas na tentativa de restaurá-la, a mensagem foi: "*Backup corrompido*. Impossível recuperar *backup*". Com receio de ter perdido todo o conteúdo do banco de dados, o administrador passou a restaurar todos os arquivos de *backups* mensais e semanais que estavam armazenados no servidor de *backup*, porém, a mensagem continuava. Ele tentou, então, recuperar o primeiro arquivo de *backup* gerado, no dia da instalação da ferramenta, quando tudo tinha sido validado. A mensagem dessa vez foi: "Iniciando processo de *Backup*. Aguarde...". Um ano de história da empresa (dados de produção, logística, recursos humanos, financeiro, administrativo, entre outros) foi perdido. O administrador de redes precisou informar o diretor da empresa que "alguns" problemas haviam ocorrido, mas que já estavam sendo resolvidos. A solução foi mandar quatro discos rígidos para uma empresa nos Estados Unidos para recuperar esses dados. O processo de recuperação durou dois meses e alguns milhares de dólares. Ainda assim, foram recuperados apenas 86% dos dados. Como não havia *backup* para restauração, um processo muito simples, muito dinheiro foi perdido e, junto com ele, 14% da história de uma empresa sólida e que investe pesadamente em tecnologia da informação para agilizar seus processos.

Um esquema de *backup* não-estruturado pode ser definido como um modelo em que não houve um planejamento de implementação, como uma pilha de DVDs com informações sobre os esquemas de *backup*. Uma desvantagem desse modelo é o seubaixo nível de recuperação dos dados, não sendo a indicação mais adequada.

VOCÊ SABIA?

Você sabia que toda semana 140.000 (cento e quarenta mil) discos rígidos tem problemas nos Estados Unidos?

O tipo de backup completo ou Full Backup é o que contém a imagem completa do sistema operacional ou dos pontos específicos que foram definidos previamente. Este tipo de *backup* tem elevado custo de armazenamento, mas é altamente necessário para os casos de falhas dos *backup* incrementais da rede. Costuma-se, como boa prática, executar um *backup* completo do sistema a cada semana ou a cada mês, dependendo da criticidade do ambiente.

O esquema de *backup* incremental e/ou diferencial é muito utilizado atualmente para a maioria dos cenários e modelos. O *backup* incremental parte de um *backup* completo realizado no sistema, e após isto, apenas as modificações dos arquivos são *backupeadas*. Isto aponta para casos de inserção de novos arquivos e ainda para modificação de arquivos que já estão backupeados, porém foram alterados e estas diferenças serão armazenadas. Este esquema tem um excelente custo/benefício e um alto grau de recuperabilidade dos *backups*.

Tão importante quanto fazer um *backup* dos dados é a maneira ou local onde os mesmos serão armazenados. Confira, no item a seguir, os meios de armazenamentos para os dados.

16.2 MEIOS DE ARMAZENAMENTO

Independente do tipo de *backup* que será realizado, os dados devem ser armazenados em algum meio de armazenamento, que geralmente são fitas magnéticas (DAT e LTO), discos rígidos (SATA, FATA, SAS) ou ainda os meios ópticos (CDs, DVDs, *Blue-Ray*). Desta forma, o administrador ou usuário deve decidir o grau de importância dos dados para escolher o melhor meio de armazenamento dos mesmos.

O armazenamento em fita magnética tem sido o meio mais comum para *backup*, arquivamento e intercâmbio de dados devido à relação custo/benefício deste meio de armazenamento. Atualmente, empresas têm desenvolvido *hardwares* específicos para *backup* em fita, como robôs inteligentes que facilitam a gerência das fitas dentro do equipamento. Hoje temos fitas magnéticas com poder de armazenamento de 3 *terabytes* sem compactação, e 6 *terabytes* compactados.

fotógrafo

Figura 39 - LTO

Os discos rígidos também são bastante utilizados para armazenamento de *backups*. A relação de capacidade/preço tem melhorado a cada ano e isto tem popularizado esse tipo de solução. Os principais motivos para adoção deste meio, são o tempo de acesso e gravação, a disponibilidade e facilidade de uso. Sobre as desvantagens, podemos citar que os HDs são facilmente danificados, principalmente em casos onde há a necessidade de transporte destes dados para outros locais.

A utilização de meios ópticos para armazenamento de *backups*, tais como CDs, DVDs e discos *Blue-ray* são geralmente usados para *desktops* e são caracterizados pelo baixo custo e baixo poder de armazenamento. A vantagem para utilização destes meios são o baixo custo relacionado que, para o usuário final, é fundamental. Entretanto, uma forte desvantagem é que os meios, desta natureza têm um curto período de vida: no caso de CDs, por exemplo, é de 5 anos, em média, e isto vai variar de acordo com a marca/qualidade da mídia.

SAIBA MAIS

Para maiores informações sobre as teorias que circundam os mecanismos de *backup*, há um excelente livro, chamado: Unix Backup & Recovery (O'REILLY, 1999).

16.3 MECANISMOS PARA *BACKUP*

Existem muitas ferramentas para realização de *backups* que vão desde ferramentas proprietárias até ferramentas distribuídas gratuitamente. Sobre as ferramentas pagas, não entraremos neste critério, visto que a ampla gama de soluções tornaria a demonstração inviável. Desta forma, na tabela a seguir, você pode visualizar uma lista com algumas ferramentas bem conhecidas para realização de *backups*, tanto de estações (*desktops*) como de servidores.

FERRAMENTA	LICENÇA	PLATAFORMA
Amanda	BSD	Windows, Linux, Mac OS X
Bacula	GPL	Windows, Linux, Mac OS X
BackupPC	GPL	Windows, Linux, Mac OS X
Cobian Backup	MPL	Windows
Mondo	GPL	Linux
DirSync Pro	GNU GPL	Windows, Linux, Mac OS X
rsync	GPL	Windows (parcialmente), Linux, Mac OS X
Duplicati	LGPL	Windows, Linux, Mac OS X

Quadro 19 - Ferramentas de *Backup*

O processo de *backup* pode ser feito por meio de *softwares* que controlam as ações para o usuários de acordo com métricas que já foram definidas previamente, como o *Amanda*, *Bacula* e *BackupPC*. Esta categoria de *softwares* trabalha na gerência de *backups*, estabelecendo a comunicação com o servidor, efetuando a gerência na transferência dos dados, criando versões do *backup* no meio de armazenamento remoto.

Além desses *softwares*, é possível realizar o *backup* de forma mais simples, usando comandos (*tar, gzip, rsync*) na linha de comando dos sistemas operacionais. Isto ainda é considerado uma forma de *backup* porque, em teoria, você estará realizando uma cópia de segurança de alguns dados que são essenciais pra você ou para a organização em que você trabalha.

Atualmente, é muito comum vermos administradores de redes, utilizando um esquema de *scritps* escritos na linguagem *shell script* em conjunto com os comandos citados no parágrafo anterior, entreoutros, para realizar o *backup*. Geralmente estes administradores utilizam a tecnologia de servidores de arquivos NFS (*Network File Systems*) para executar remotamente estes comandos e terem os dados a salvo em um Servidor de *Backup* de fato. Um exemplo de arquitetura de servidores de *backup* pode ser visualizado, na figura a seguir.

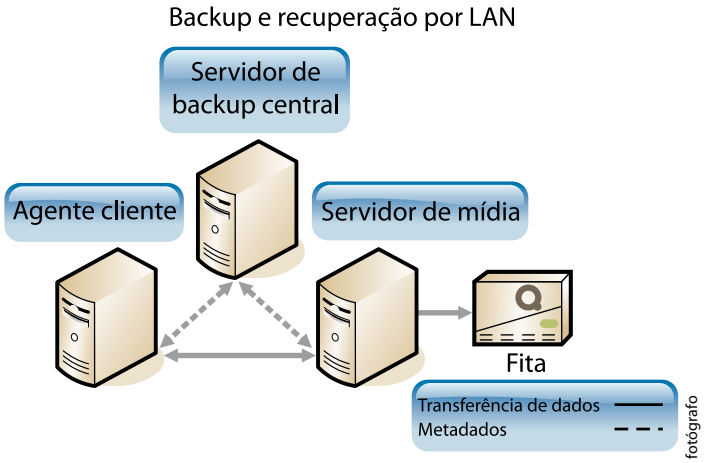


Figura 40 - Exemplo de Arquitetura de *Backup*

16.4 INSTALAÇÃO DE UMA FERRAMENTA DE *BACKUP*

Como exemplo de instalação de um servidor de backups, usaremos o *rsync* sob um sistema operacional Debian Linux/GNU na versão *Squeeze*. Entretanto, é importante salientar que o *rsync* é compatível com uma variedade de sistemas operacionais, incluindo Windows, Linux e Mac OS X, atendendo, dessa forma, uma grande variedade de tipos de servidores. O *rsync* é um protocolo de rede e aplicação para sistemas heterogêneos que tem a função de sincronizar arquivos e diretórios de um local para outro. No modo *daemon* ele escuta conexões na porta 873 (TCP) por meio do protocolo *rsync* ou ainda, pode servir arquivos via remote *shell*, RSH e SSH. O *rsync* é um *software* livre, licenciado pela licença GNU, inventado por Andrew Tridgell (mesmo inventor do Samba) and Paul Mackerras, em 1996.

No exemplo a seguir, é possível visualizar a instalação do *rsync* no sistema operacional Debian, via gerenciador de pacotes Debian (APT).

```
# apt-get install rsyncd
```

Após a instalação, é possível fazer inúmeras customizações de funcionalidade e segurança que devem estar alinhadas à nossa política de *backup*. Em linhas gerais, exceto pela parte de autenticação que dependerá do método e protocolo utilizados, nós temos que configurar o sistema que servirá de servidor de *backup* e configurar o cliente, para que ele nos envie os arquivos. Veja, a seguir, um pequeno exemplo de configuração de um servidor *rsync*.

```
# Exemplo de Configuração do rsyncd.conf
#
# OPÇÕES GLOBAIS
#

motd file=/etc/motd
log file=/var/log/rsyncd.log
pid file=/var/run/rsyncd.pid
syslog facility=daemon
uid=0

#
# MODULE OPTIONS
#

[Arquivos]
path=/home/arquivos
comment = Diretório de Backup dos Arquivos
read only = yes
max connections=10
transfer logging = yes
log format = %t: host %h (%a) %o %f (%l bytes). Total
%b bytes.
```

Após a instalação e configuração do servidor de *backup* (*rsyncd*), e depois de elaboradas as customizações de funcionalidades e segurança da operação de *backup*, é hora de configurar o cliente para enviar os arquivos para o servidor. Isso, na maioria das vezes, é feito com *softwares* escritos em *shell script*. Um exemplo de um destes pode ser visualizado a seguir.

```
#!/bin/sh
#
# Nome: backupArquivos.sh
# Descrição: Script para backup dos arquivos em /home/
arquivos/
#
# Ajuste de formato de data

DATA=`date +%Y-%m-%d_%H:%M`

# Ajuste do diretório
cd /rsync/samba
rsync -av --backup --backup-dir=$DATA 192.168.1.1::ar-
quivos

#
# EOF
```

Após a configuração do cliente, é importante criar o arquivo de senhas para autenticação, se este for o caso. Mais uma observação importante é a utilização de *crontab* para agendamento da rotina de *backup*. No crontrab, é possível definir a política de *backup* que irá gerar, ou seja, sua periodicidade.

Veja um exemplo da linha do /etc/crontab.

```
0 1 * * * root /bin/backupArquivos.sh
```

Nesse exemplo, o processo de *backup* iniciará todos os dias, a 1:00 hora da manhã, copiando todos os arquivos do "/home/arquivos". Tenha sempre em mente queo esquema de *backup* será utilizado para sincronizar as políticas com as necessidades reais que a situação exige.

FIQUE ALERTA

A tarefa de realização de *backup* é muito importante, como você pôde ver durante o capítulo, porém, fazer testes de recuperação do *backup* são tão importantes quanto.

RECAPITULANDO

Neste capítulo, você conheceu aspectos relacionados aos servidores de *backup*, estudou os tipos de *backup* mais populares como o não-estruturado, completo e incremental/diferencial. Após esta etapa, você estudou os meios mais comuns de armazenamento de *backup*, que são as fitas magnéticas, os discos rígidos e os mecanismos ópticos tais como DVDs e CDs e, para finalizar, viu um exemplo de instalação usando o *rysnc* para transferência e sincronia de *backup*.

E nossa viagem ao mundo dos serviços de redes chega ao fim. Agora é com você! Lembre-se de que, para ter sucesso na carreira, é preciso estar sempre bem informado, ter certeza do que está fazendo e procurar conhecer sempre as novidades na área. Dedicação também é uma boa ferramenta que o administrador de redes deve ter sempre consigo. Sucesso na sua escolha!

Anotações:

REFERÊNCIAS

APACHE WEB SERVER. Disponível em: <http://httpd.apache.org>. Acesso em: 14 set. 2011.

BIND (2011): BIND. Disponível em: <http://en.wikipedia.org/wiki/BIND>. Acesso em: 14 set. 2011.

CERN. Disponível em: <http://public.web.cern.ch/public/en/about/web-en.html>. Acesso em: 14 set. 2011.

GEEK. **A origem do @.** Disponível em: <http://www.geek.com.br/posts/9921-a-origem-do>. Acesso em: 16 set. 2011.

HUNT, Craig. **Linux:** servidores de Rede. Rio de Janeiro: Ciência Moderna, 2004.

MAIL SERVER (2011): MAIL SERVER, WIKIPÉDIA. Disponível em: <http://en.wikipedia.org/wiki/Email>. Acesso em: 14 set. 2011.

NEMETH, Evi; SNYDER, Garth; HEIN, Trent R. **Manual Completo do Linux:** Guia do Administrador. 2. ed. São Paulo: Pearson Pretince Hall, 2007.

NETCRAFT. Disponível em: <http://news.netcraft.com/archives/2011/03/09/march-2011-web-server-survey.html>. Acesso em: 14 set. 2011.

NEWS GENERATION. **Rede Nacional de Ensino e Pesquisa - RNP.** Disponível em: <http://www.rnp.br/newsgen/0011/ftp-passivo.html>. Acesso em: 17 set. 2011.

OPENLDAP. Disponível em: <http://www.openldap.org>. Acesso em: 16 set. 2011.

OPENSSL. Disponível em: <http://www.openssl.org>. Acesso em: 14 set. 2011.

O´REILLY (1999): PRESTON, W. Curtis. Unix Backup and Recovery. O'Reilly and Associates, 1999.

POSTFIX. Disponível em: <www.postfix.org>. Acesso em: 15 set. 2011.

PRESTON, W. Curtis. **Unix Backup and Recovery**. [S.l.]: O'Reilly and Associates, 1999.

PROCMAIL. Disponível em: <www.procmail.org>. Acesso em: 15 set. 2011.

RNP (2011): FTP, RNP. Disponível em: <http://www.rnp.br/newsgen/0011/ftp-passivo.html>. Acesso em: 17 set. 2011.

ROOT-SERVER. Disponível em: <http://www.root-servers.org>. Acesso em: 10 set. 2011.

SECURITY SPACE. **Mail MX Survey.** Disponível em: <http://www.securityspace.com/s_survey/data/man.201007/mxsurvey.html>. Acesso em: 15 set. 2011.

SMITH, Roderick W. **Redes Linux Avançadas.** Rio de Janeiro: Ciência Moderna, 2003.

SQUID CACHE. Disponível em: <http://www.squid-cache.org/>. Acesso em: 13 set. 2011.

WEB SERVER (2011): WEB SERVER. **Apache Web Server**. Disponível em: <http://httpd.apache.org>. Acesso em: 14 set. 2011.

WIKIPÉDIA. **Common Internet File System - CIFS.** Disponível em: <http://en.wikipedia.org/wiki/CIFS>. Acesso em: 18 set. 2011.

WIKIPÉDIA. **Domain Information Groper - DIG.** Disponível em: <http://en.wikipedia.org/wiki/Domain_Information_Groper>. Acesso em: 10 set. 2011.

WIKIPÉDIA. **List Mail Server.** Disponível em: <http://en.wikipedia.org/wiki/List_of_mail_servers>. Acesso em: 15 set. 2011.

WIKIPÉDIA. **Mail Server**. Disponível em: <http://en.wikipedia.org/wiki/Email>. Acesso em: 14 set. 2011.

WIKIPÉDIA. **Server Message Block.** Disponível em: <http://en.wikipedia.org/wiki/Server_Message_Block>. Acesso em: 17 set. 2011.

WIKIPÉDIA. **Web Servers**. Disponível em: <http://en.wikipedia.org/wiki/Web_server>. Acesso em: 14 set. 2011.

MINICURRÍCULO DOS AUTORES

Douglas D. J. de Macedo é aluno de doutorado do Programa de Pós-Graduação de Engenharia do Conhecimento (PPGEGC) da Universidade Federal de Santa Catarina (UFSC), programa no qual já tem o título de Mestre (M.Eng.). Tem graduação de Tecnologia em Processamento de Dados pela FATEC e é Especialista em Segurança de Redes de Computadores pela Faculdade SENAI/Florianópolis. Profissionalmente, atua como gerente de redes na operação da Rede Catarinense de Telemedicina (RCTM) e é pesquisador associado no Instituto Nacional de Convergência Digital (INCoD) da UFSC. É professor do Curso Superior em Redes de Computadores do SENAI/Florianópolis, ministrando as disciplinas de Serviços de Redes e Integração de Serviços de Redes.

ÍNDICE

SENAI - DN
UNIDADE DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA – UNIEP

Diana Neri
Gestora do Projeto Estratégico de Recursos Didáticos Nacionais

Paula Martini
Gestora do Programa Nacional de Oferta de Educação Profissional na Modalidade a Distância

SENAI - DEPARTAMENTO REGIONAL DE SANTA CATARINA

Simone Moraes Raszl
Beth Schirmer
Coordenação EaD

Carolinaxxxxxxxxxxxxx
Coordenação Projetos EaD

xxxxxxxxxxxxxxx
Coordenação Técnica

Gisele Umbelino
Coordenação de Desenvolvimento de Recursos Didáticos

Douglas D. J. de Macedo
Autor

Evelin Lediani Bao
Design Educacional

Sidiane Kayser dos Santos Schwinzer
Revisão Normativa

Sidiane Kayser dos Santos Schwinzer
Revisão Ortográfica e Gramatical

D'imitre Camargo Martins
Diego Fernandes
Luiz Eduardo Meneghel
Ilustrações, Tratamento de Imagens

Daniela de Oliveira Costa
Diagramação

Juliana Vieira de Lima
Revisão e Fechamento de Arquivos

Daiani Machado
Apoio técnico

Patrícia Correa Ciciliano - CRB-14/752
Bibliotecária - Ficha Catalográfica

---

i-Comunicação
Projeto Gráfico

SENAI

Iniciativa da CNI - Confederação Nacional da Indústria

ISBN 978-85-7519-484-3