GNU/LINUX **agosto/2009**

Comandos básicos/comandos de rede

1 - LINUX: DEFINIÇÃO

O sistema operacional Linux é de código-fonte aberto. Esse sistema foi criado por Linus Torvalds, em 1991, enquanto ele ainda era estudante de Ciência da Computação na Universidade de Helsinque, Finlândia.

Linus criou o Linux a partir do Minix, um sistema operacional também de código fonte aberto criado pelo professor Andrew S. Tanenbaum em 1987. A proposta do Minix era ser um clone aberto do Unix.

O Linux é aderente ao padrão ou comitê POSIX [Portable Operating System Interface for Unix, de 1988]. Todo sistema aderente ao POSIX é dito membro da família Unix.

Os principais membros dessa família atualmente são: Solaris [Sun Microsystems], HP-UX [HP], AIX [IBM] e Linux [código fonte aberto].

O Linux não o único membro da família Unix de código fonte aberto, como exemplo temos também o FreeBSD, da Berkeley Software Distribution.

A licença de uso do Linux é GNU General Public License.

Sabemos hoje que o grande mérito de Linus Torvalds não foi a criação do Linux, mas principalmente o modelo aberto e colaborativo que ele inaugurou pela internet para conseguir resolver os problemas do sistema. Esse modelo colaborativo foi posteriormente copiado por outros grupos de desenvolvimento.

Tecnicamente, Linux é apenas um kernel [núcleo do sistema operacional]. O código fonte está disponível, sem custo, em http://www.kernel.org.

Compilar um kernel para determinada arquitetura de computador não é tarefa trivial. Por isso, as distribuições Linux vieram para simplificar a vida do usuário, e também para facilitar a tarefa de divulgação do Linux[1].

2 - DISTRIBUIÇÃO LINUX

Por definição, distribuição Linux é uma empresa que junta o kernel Linux, aplicações,

1 - O que nem sempre é verdade, pois algumas distribuições são tão amadoras que denigrem a imagem do pinguim

utilitários e um instalador amigável num pacote a ser distribuído livremente. O objetivo comercial é vender licenças de suporte. Nesse pacote, geralmente é incluído apenas software de código fonte livre, porém em algumas distribuições encontramos inclusive software proprietário.

Exemplos de distribuições Linux:

- Red Hat [www.redhat.com], é voltada para o segmento servidor. A versão para desktop é o Fedora.

- Suse [www.novell.com/linux], é voltada para o segmento servidor. Para entrar nesse segmento, a Novell comprou a distribuição Suse.

- Mandriva [www.mandriva.com] é voltada para o segmento desktop. Essa distribuição surgiu da fusão da Mandrake [França] com a Conectiva [Brasil].

- Debian [www.debian.org] voltada para o segmento desktop. Tem a tradição de não incluir software proprietário.

- Slackware [www.slackware.com] voltada para o segmento desktop. Tem a tradição de usar muito a linha de comando.

3 - LINHA DE COMANDO

A única interface comum a todos os sistemas membros da família Unix é a linha de comando, portanto toda a administração desses sistemas é feita a partir de comandos.

Os comandos são disparados de uma aplicação chamada shell. O shell também é conhecido por interpretador de comandos. Ao contrário do DOS/Windows, no mundo Unix existem dezenas de interpretadores de comandos, por exemplo:

bash: Bourne Shell Again
ksh: Korn Shell
csh: C shell
sh: Bourne Shell

Para a linha de comando, basicamente todo shell é igual. Porém, para escrever shell script precisa usar a linguagem específica de cada shell.

4 - EXEMPLOS DE COMANDOS

Notação usada nesses exemplos:

```
o comando:              shell$ ls
a saida do comando      arquivo1    dir3
```

Para saber onde se encontra no sistema hierárquico de arquivo:

```
shell$ pwd
/home/aluno
```

Comando para se deslocar no sistema hierárquico de arquivo:

```
shell$ cd  /etc
shell$ pwd
/etc
```

Listagem do diretório /bin

```
shell$ ls  /bin
alsaunmute      dbus-monitor    false       link        nice            rvi         tracepath6
arch            dbus-send       fgrep       ln          nisdomainname   rview       traceroute
awk             dbus-uuidgen    find        loadkeys    ntfs-3g         sed         traceroute6
basename        dd              fusermount  login       ntfs-3g         probe       setfont
....
```

Listagem no formato longo:

```
shell$ ls  -l  /bin
-rwxr-xr-x.   1  root  root      123 Mai  15 09:38 alsaunmute
-rwxr-xr-x.   1  root  root    36608 Abr   1 08:20 arch
lrwxrwxrwx. 1  root  root        4 Jun  14 15:19 awk -> gawk
-rwxr-xr-x.   1  root  root    34720 Abr   1 08:20 basename
-rwxr-xr-x.   1  root  root  838824 Abr   8 07:46 bash
-rwxr-xr-x.   1  root  root    57096 Abr   1 08:20 cat
-rwxr-xr-x.   1  root  root    63804 Abr   1 08:20 chgrp
-rwxr-xr-x.   1  root  root    59652 Abr   1 08:20 chmod
-rwxr-xr-x.   1  root  root    66608 Abr   1 08:20 chown
-rwxr-xr-x.   1  root  root  110996 Abr   1 08:20 cp
....
```

Mostrar o conteúdo de arquivo de texto:

```
shell$ more  /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
vcsa:x:69:69:virtual console memory owner:/dev:/sbin/nologin
avahi-autoipd:x:499:499:avahi-autoipd:/var/lib/avahi-autoipd:/sbin/nologin
oprofile:x:498:498:Special user account to be used by OProfile:/home/oprofile:/s
bin/nologin
ntp:x:38:38::/etc/ntp:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
polkituser:x:87:87:PolicyKit:/:/sbin/nologin
avahi:x:497:497:avahi-daemon:/var/run/avahi-daemon:/sbin/nologin
--Mais--(51%)
```

Pegar e mostrar todo o conteúdo de um arquivo de uma única vez:

```
shell$ cat  /etc/inittab
# inittab is only used by upstart for the default runlevel.
# ADDING OTHER CONFIGURATION HERE WILL HAVE NO EFFECT ON YOUR SYSTEM.
# System initialization is started by /etc/event.d/rcS
# Individual runlevels are started by /etc/event.d/rc[0-6]
# Ctrl-Alt-Delete is handled by /etc/event.d/control-alt-delete
# Terminal gettys (tty[1-6]) are handled by /etc/event.d/tty[1-6] and
# /etc/event.d/serial
# For information on how to write upstart event handlers, or how
# upstart works, see init(8), initctl(8), and events(5).
# Default runlevel. The runlevels used are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single user mode
#   2 - Multiuser, without NFS (The same as 3, if you do not have networking)
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:
```

Busca expressão regular no conteúdo do arquivo:

```
shell$ grep  operator  /etc/passwd
operator:x:11:0:operator:/root:/sbin/nologin
```

Com a opção **-v**, **grep** faz uma busca reversa, para ignorar todas as linhas que contenham o padrão de busca:

```
shell$ grep  -v  "#"  /etc/inittab
id:5:initdefault:
```

Mostra as 10 últimas linhas de um arquivo:

```
shell$ tail   /etc/group
stapdev:x:491:
stapusr:x:490:
wbpriv:x:88:squid
smolt:x:489:
torrent:x:488:
haldaemon:x:68:
squid:x:23:
gdm:x:42:
aluno:x:500:
jackuser:x:487:
```

Mostra as 4 últimas linhas de um arquivo:

```
shell$ tail  -4  /etc/group
squid:x:23:
gdm:x:42:
aluno:x:500:
jackuser:x:487:
```

NOTA: **tail** com opção **-f** mostra as últimas linhas de um arquivo e permanece tentando ler novas linhas, à medida que forem sendo escritas. Isso é muito útil para acompanhar acessos em arquivos de logs. Exemplo: **tail -f /var/log/messages**

Mostra as lnhas do início de um arquivo:

```
shell$ head  /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
```

Conta o número de linhas, palavras e caracteres de um arquivo:

```
shell$ wc  /etc/passwd
40   65  1986  /etc/passwd
```

que tem 40 linhas, 65 palavras e 1986 caracteres.

Determina o tipo de um arquivo:

```
shell$ file  /etc/group
/etc/group: ASCII text

shell$ file  /bin/ls
/bin/ls: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked (uses shared libs),
for GNU/Linux 2.6.18, stripped

shell$ file  /usr
/usr: directory
```

Copiar um arquivo:

```
shell$ cp  /etc/passwd  /tmp
```

Mover um arquivo:

```
shell$ mv  /tmp/passwd  /tmp/passwd2
```

Criar diretório:

```
shell$ mkdir  /home/aluno/teste
```

Remover diretório vazio:

```
shell$ rmdir  /home/aluno/teste
```

Remover arquivo regular:

```
shell$ rm  /tmp/passwd2
```

NOTA:
para remover diretório com conteúdo, usar o comando **rm** com opção **-r** [recursivo]
Exemplo: **rm -r /home/aluno/teste**

5 - PIPES

Os pipes são implementações de comunicação entre processos. Na linha de comando, se for disparado mais de um executável de uma vez, vai gerar mais de um processo. E se esses executáveis estiverem ligados por um pipe "|", então a saída de um processo será a entrada do outro processo. Essa sequência de comunicação vai da esquerda para a direita.

Exemplo:
ls /bin | more

onde **ls /bin** lista o conteúdo do diretório /bin, e como essa lista não cabe nas dimensões da janela shell, então é usado na mesma linha de comando o executável "**more**" para apresentar essa listagem paginadamente.

Outro exemplo:

cat /etc/passwd | grep bash | wc -l

que determina o número de usuário cadastrados no sistema que têm o shell /bin/bash.

6 - REDIRECIONAMENTOS

Os redirecionamentos alteram a entrada ou saída padrão de um comando. Normalmente, a entrada padrão é o teclado, e a saída padrão é o monitor.

Saída padrão:

Por exemplo, o comando **ls /etc > /tmp/ls_etc.txt**

faz uma listagem do diretório **/etc** e envia a saída para o arquivo **/tmp/ls_etc.txt**. Se o arquivo não existisse, seria criado. No entando, se o arquivo existisse, o conteúdo anterior seria perdido.

Caso o arquivo existisse, para manter o conteúdo anterior e acrescentar o novo conteúdo no final do arquivo, o comando seria **ls /etc >> /tmp/ls_etc.txt**

NOTA: quando se redireciona a saída padrão, esta vai para um arquivo e, portanto, não tem saída no shell [não vem saída para o monitor].

Saída de erro:

Caso o comando encontrasse algum erro, retornaria uma mensagem de erro. Por exemplo, tentar criar um diretório que já existe gera mensagem de erro. Essa mensagem de erro é chamada de saída de erro, e para ser redirecionada precisa incluir o número 2 antes do sinal de maior ">".

mkdir /etc 2> /tmp/erro_do_comando.txt

NOTA: se não usar essa notação a saída de erro não é redirecionada e vem para o monitor.

Entrada padrão:

A entrada padrão é redirecionada com o sinal "<" após o comando. Com isso, o comando irá ler a entrada de um arquivo e não mais do teclado.

Por exemplo, considere o executável que, ao ser disparado pergunta pelo nome e idade, depois finaliza informando o nome e idade:

```
shell$ nome_idade
        entre com o seu nome: juca
        entre com a sua idade: 286
Seu nome é juca, sua idade é 286 anos
```

Se esse comando fosse disparado com redirecionamento da entrada padrão, não esperaria pelas entradas de teclado.

```
shell$ nome_idade  <  arq_entrada
        entre com o seu nome:
        entre com a sua idade:
Seu nome é juca, sua idade é 286 anos
```

No arquivo arq_entrada existem apenas duas linhas, que são as entradas pedidas pelo comando:
--- arq_entrada ---
juca
286

Desse modo, existem 3 casos para os redirecionamentos:

- saída padrão > [o correto seria 1>, porém pode ser omitido o número 1]
- saída de erro 2>
- entrada padrão <

A forma clássica usada no shell para redirecionar de uma única vez tanto saída de erro quanto saída padrão para um arquivo, é:

comando > arquivo_saida 2>&1

7 - EDITOR vi/vim

No mundo Unix/Linux, existe apenas um editor de texto realmente universal: é o vi [visual interface]. Embora não seja amigável ao iniciante, vale o esforço para aprender pois em qualquer sistema membro da família Unix ele está presente.

É certo que existem muitos editores, alguns até mais práticos que o vi. Porém, não há a garantia de que ele esteja instalado na máquina que você for administrar.

O editor vim é o vi "improved" [aprimorado], e deve ser usado sempre que estiver disponível, pois é mais amigável que o vi. Quanto aos comandos, são idênticos.

A primeira noção que se precisa ter do vi/vim é que se trata de um editor de 2 modos, que são edição ou comando. Afinal, vi/vim usa apenas o teclado, então num caso teclar estará editando o texto de algum arquivo, no outro estará dando comandos [por exemplo, para salvar o texto].

Para entrar no modo inserção, basta pressionar a letra **i** [de insert]. A partir daí, se está escrevendo no arquivo.

Para sair do modo inserção e entrar no modo comando, deve comandar a sequência <**ESC**> <**SHIFT**> <**:**>, que são as 3 teclas **ESC+SHIFT+:** e posteriormente pressionar a tecla [ou teclas] com o comando. Nessa situção, a tecla **w** [write] salva o texto no arquivo, a tecla **q** [quit] abandona o editor de texto e volta para o shell.

Exemplo: para criar o arquivo teste.txt e escrever no seu interior, basta seguir a sequência:

1) no shell, comandar "**vi teste.txt**", que abre o editor e toma o shell;
2) Pressionar a tecla "**i**" para entrar no modo de inserção de texto;
3) Escrever o texto "teste de escrita vi";
4) Sair do modo de inserção e entrar no modo comando com a sequência de teclado <**ESC**> <**SHIFT**> <**:**>
5) pressionar as teclas **wq**, que vai salvar e sair do editor.

Os principais caracteres usados no modo comando são:

x deleta o carater onde está o cursor;
dd apaga a linha onde está o cursor;
u desfaz a edição;
r substitui o caracter na posição do cursor pelo que for teclado após o r.

8 - SCRIPTS

Além da linha de comando, o administrador do sistema costuma usar scripts [roteiros] para automatizar tarefas rotineiras. Desse modo, ao invés de disparar a mesma sequência de comandos todo dia, basta escrever essa sequência num arquivo de texto, definir na primeira linha o nome do interpretador, colocar permissão para execução e disparar o script.

Exemplo 1:

```
----- teste.sh -------
#!/bin/bash
echo "Script teste"
----------------------
```

Dar permissão de execução:

```
shell$ chmod  755  teste.sh
```

Disparar o executável script:

```
shell$ ./teste.sh
Script teste
```

Exemplo 2:

```
--- nome_idade ------------------------------------
#!/bin/bash
echo -n "Escreva o seu nome: "
read nome
echo -n "Escreva a sua idade: "
read idade
echo "Seu nome é $nome, sua idade é $idade"
---------------------------------------------------------
```

Dar permissão de execução:

```
shell$ chmod  755  nome_idade
```

Executar o script:

```
shell$ ./nome_idade
Escreva o seu nome: juca
Escreva a sua idade: 286
Seu nome é juca, sua idade é 286
```

9 - COMANDOS DE REDES

Mostrar o dispositivo de rede [hardware]: **lspci**

```
shell$ lspci | grep -i ethernet
00:0f.0 Ethernet controller: nVidia Corporation MCP73 Ethernet (rev a2)
```

Mostrar a configuração lógica da(s) interface(s) de rede(s): **ifconfig**

```
shell$ ifconfig
eth0      Link encap:Ethernet  Endereço de HW 00:21:97:80:7C:FF
          inet end.: 192.168.1.10  Bcast:192.168.1.255  Masc:255.255.255.0
          endereço inet6: fe80::221:97ff:fe80:7cff/64 Escopo:Link
          UP BROADCASTRUNNING MULTICAST  MTU:1500  Métrica:1
          RX packets:55602 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10620 errors:0 dropped:0 overruns:0 carrier:0
          colisões:0 txqueuelen:1000
          RX bytes:8883790 (8.4 MiB)  TX bytes:2304141 (2.1 MiB)
          IRQ:28 Endereço de E/S:0x6000

lo        Link encap:Loopback Local
          inet end.: 127.0.0.1  Masc:255.0.0.0
          endereço inet6: ::1/128 Escopo:Máquina
          UP LOOPBACKRUNNING  MTU:16436  Métrica:1
          RX packets:18 errors:0 dropped:0 overruns:0 frame:0
          TX packets:18 errors:0 dropped:0 overruns:0 carrier:0
          colisões:0 txqueuelen:0
          RX bytes:1268 (1.2 KiB)  TX bytes:1268 (1.2 KiB)
```

Mostrar as configurações da interface de rede [como root]: **ethtool**

```
# ethtool  eth0
Settings for eth0:
        Supported ports: [ MII ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supports auto-negotiation: Yes
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
        Advertised auto-negotiation: Yes
        Speed: 100Mb/s
        Duplex: Full
        Port: MII
        PHYAD: 1
        Transceiver: external
        Auto-negotiation: on
        Supports Wake-on: g
        Wake-on: d
        Link detected: yes
```

Mostrar o conteúdo da tabela de rotas: **route**

```
shell$ route -n
Tabela de Roteamento IP do Kernel
Destino         Roteador        MáscaraGen.     Opções Métrica Ref    Uso Iface
192.168.1.0     0.0.0.0         255.255.255.0   U      1       0      0   eth0
0.0.0.0         192.168.1.1     0.0.0.0         UG     0       0      0   eth0
```

Os comandos, ifconfig, ethtool e route também são usados para reconfigurar a rede. Por exemplo:

```
# ifconfig eth0 192.168.1.10 netmask 255.255.255.0 up
```

configura o endereço 192.168.1.10/255.255.255.0 na interface eth0

```
# route add default gw 192.168.1.1 eth0
```

configura a rota default 192.168.1.1 para a interface eth0

Para saber se o equipamento está acessando a rede, use o comando **ping**. Por exemplo:

```
shell$ ping localhost
PING localhost (127.0.0.1) 56(84) bytes of data.
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.076 ms
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.045 ms
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=64 time=0.041 ms
64 bytes from localhost (127.0.0.1): icmp_seq=4 ttl=64 time=0.039 ms
^C
--- localhost ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3308ms
rtt min/avg/max/mdev = 0.039/0.050/0.076/0.015 ms

shell$ ping 192.168.1.20
 ping 192.168.1.20
PING 192.168.1.20 (192.168.1.20) 56(84) bytes of data.
64 bytes from 192.168.1.20: icmp_seq=1 ttl=64 time=38.6 ms
64 bytes from 192.168.1.20: icmp_seq=2 ttl=64 time=9.33 ms
64 bytes from 192.168.1.20: icmp_seq=3 ttl=64 time=12.0 ms
64 bytes from 192.168.1.20: icmp_seq=4 ttl=64 time=10.3 ms
64 bytes from 192.168.1.20: icmp_seq=5 ttl=64 time=19.1 ms
^C
--- 192.168.1.20 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4846ms
rtt min/avg/max/mdev = 9.337/17.902/38.698/10.948 ms
```

Para resolver o IP de determinado host, usar o comando **nslookup** ou **dig**:

```
shell$ nslookup  www.jairo.pro.br
Server:         201.91.128.194
Address:        201.91.128.194#53

Non-authoritative answer:
Name:   www.jairo.pro.br
Address: 187.16.23.139

shell$ ping 187.16.23.139
PING 187.16.23.139 (187.16.23.139) 56(84) bytes of data.
64 bytes from 187.16.23.139: icmp_seq=1 ttl=57 time=12.1 ms
64 bytes from 187.16.23.139: icmp_seq=2 ttl=57 time=9.24 ms
64 bytes from 187.16.23.139: icmp_seq=3 ttl=57 time=13.1 ms
^C
--- 187.16.23.139 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2852ms
rtt min/avg/max/mdev = 9.249/11.525/13.194/1.666 ms
```

```
shell$ dig  www.jairo.pro.br

; <<>> DiG 9.6.1b1-RedHat-9.6.1-0.3.b1.fc11 <<>> www.jairo.pro.br
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62325
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 1

;; QUESTION SECTION:
;www.jairo.pro.br.              IN      A

;; ANSWER SECTION:
www.jairo.pro.br.       3111    IN      A       187.16.23.139

;; AUTHORITY SECTION:
jairo.pro.br.           3111    IN      NS      ns1.hostnet.com.br.
jairo.pro.br.           3111    IN      NS      ns2.hostnet.com.br.

;; ADDITIONAL SECTION:
ns1.hostnet.com.br.     2024    IN      A       200.185.109.63

;; Query time: 56 msec
;; SERVER: 201.6.0.115#53(201.6.0.115)
;; WHEN: Sun Aug 23 16:53:53 2009
;; MSG SIZE  rcvd: 114
```

Para ver conexões de rede, tabelas de roteamento, estatísticas de interface e conexões mascaradas, use o comando **netstat**:

```
shell$ netstat -na | more
Conexões Internet Ativas (servidores e estabelecidas)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:22              0.0.0.0:*               OUÇA
tcp        0      0 127.0.0.1:631           0.0.0.0:*               OUÇA
tcp        0      0 192.168.1.10:47311      64.233.163.19:80        ESTABELECIDA
tcp        0      0 192.168.1.10:47312      64.233.163.19:80        ESTABELECIDA
tcp        0      0 ::1:631                 :::*                    OUÇA
udp        0      0 0.0.0.0:47061           0.0.0.0:*
udp        0      0 0.0.0.0:5353            0.0.0.0:*
udp        0      0 0.0.0.0:631             0.0.0.0:*
udp        0      0 0.0.0.0:68              0.0.0.0:*
Domain sockets UNIX ativos (servidores e estabelecidas)
--Mais--
```

Para descobrir quais serviços determinado host disponibiliza, use o comando **nmap**, que faz um scan de portas:

```
# nmap  localhost

Starting Nmap 4.76 ( http://nmap.org ) at 2009-08-23 17:02 BRT
Warning: Hostname localhost resolves to 2 IPs. Using 127.0.0.1.
Interesting ports on localhost (127.0.0.1):
Not shown: 998 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
631/tcp open  ipp
```

```
# nmap  192.168.1.20

Starting Nmap 4.76 ( http://nmap.org ) at 2009-08-23 17:02 BRT
Warning: Hostname localhost resolves to 2 IPs. Using 127.0.0.1.
Interesting ports on localhost (127.0.0.1):
Not shown: 998 closed ports
PORT    STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
80/tcp    open  http
3128/tcp open  squid-http
```

10 - INSTALAÇÃO DE PACOTES

A distribuição Linux que temos no laboratório é derivada do Debian, portanto o gerenciamento de pacotes, para instalação, é feito com o comando **apt-get**.

O comando **apt-get** lê o arquivo de configuração **/etc/apt/sources.list** para determinar os locais com repositório de pacotes para instalação.

Normalmente, a configuração em **sources.list** está correta, porém com frequência dá erro na instalação pois aparece na primeira linha uma configuração do tipo:

deb cdrom:[Debian GNU/Linux 5.0.1 _Lenny_ - Official i386 DVD Binary-1 20090413-00:33]/ lenny contrib main

indicando que **apt-get** está configurado para buscar pacotes na mídia de cd-rom, mas que não se encontra no drive no momento.

Se este for o caso, a solução então é editar o arquivo **sources.list** e comentar essa linha [colocar uma cerquilha "#" na frente da linha]:

 # deb cdrom:[Debian GNU/Linux 5.0.1 _Lenny_ - Official i386 DVD Binary-1 20090413-00:33]/ lenny contrib main

Depois disso é necessário atualizar o database de repositórios com o comando:

```
# apt-get  update
```

Agora, o comando para instalação de pacotes deverá funcionar.

Por exemplo, para instalar o pacote **nmap**, comandar:

```
# apt-get  install  nmap
```

NOTA: para instalar pacotes precisa ser root. O jeito simples de adquirir poderes de root é com o comando sudo. Por exemplo:

```
shell$ sudo  su
[sudo] password for aluno:
# id
uid=0(root) gid=0(root) grupos=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)
```

após enviar a senha de aluno, ganha o shell de root, como pode ser verificado com o símbolo do shell que agora é uma cerquilha "#".