



Instalar programas a partir do fonte

Sumário

Capítulo 1

Instalar programas a partir do fonte	3
1.1. Mãos a obra.....	4

Capítulo 2

Gerenciando	5
2.1. Objetivos.....	5
2.1. Troubleshooting.....	5

Índice de tabelas

Índice de Figuras

Capítulo 1

Instalar programas a partir do fonte

- Descompactar o código fonte usando a compressão comum;
- Princípios básicos de invocação fazer programas para compilar;
- Aplicar parâmetros para um script de configuração;
- Saber onde as fontes são armazenados por padrão.

1.1. Mãos a obra

Para instalar programas em distribuições GNU/Linux, o usuário pode usar diversas maneiras com por exemplo, gerenciadores de pacotes de baixo e alto nível. Mas é através do código fonte que podemos personalizar a instalação desses programas, assim melhorando a segurança e aumentando sua performance. O processo se chama compilação:



Mas afinal o que é compilação?

É um processo de transformação de um programa em código fonte, que é escrito por um programador, em uma linguagem de máquina que é um programa binário, ou se já executável.



Mas quem é o responsável em fazer a compilação?

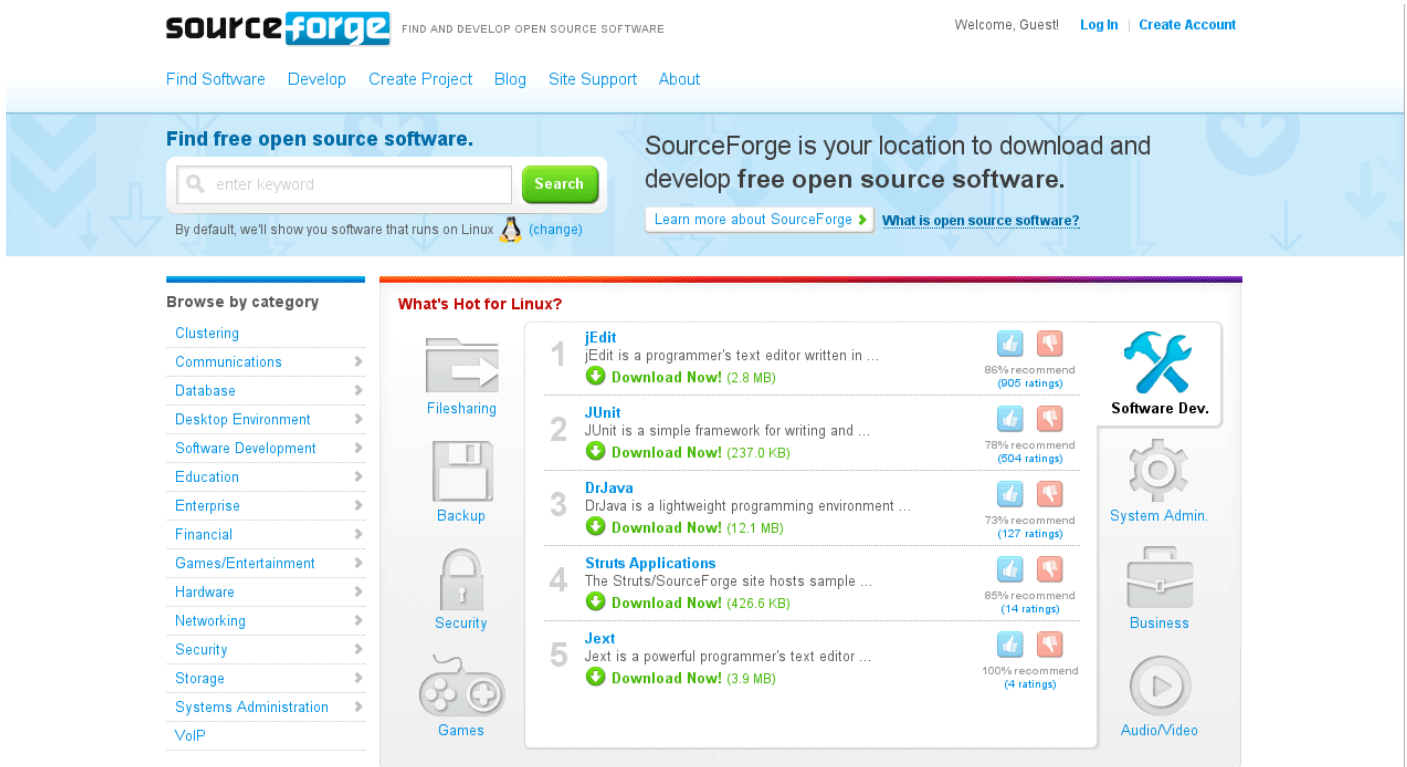
O compilador que é um programa ou um grupo de programas, a partir de um código fonte consegue gerar o binário.

Para iniciar todo esse processo é necessário obter o programa através do código fonte. Uma das maneiras é acessar o site do projeto e fazer download do programa, que pode vir empacotado e compactado com tar gzip e/ou tar bzip2. Um site que você pode encontrar muitos projetos é o sourceforge.

Saiba mais acessando:



<http://sourceforge.net>



Como exemplo pratico, vamos instalar o Wireshark pelo código fonte, mas antes de mais nada instale os compiladores que serão necessários para a compilação:



```
# aptitude install build-essential
```

Para iniciar nossa compilação, faça o download do wireshark no terminal através do comando wget:



```
# wget -c http://ufpr.dl.sourceforge.net/project/wireshark/src/all-versions/wireshark-1.2.10.tar.bz2
```

Para continuar descompacte o wireshark em /usr/local



```
# tar -xjvf wireshark-1.2.10.tar.bz2 -C /usr/local
```

Vamos entender as flags usadas com o comando tar:

x - Extrai o conteúdo do arquivo;

v - Executa a extração no modo verbose;

j - Usado para arquivos com extensão bz2 (bzip2);

f - Indica qual arquivo será descompactado;

C - Indica o caminho onde o arquivo será descompactado.



Porque descompactar em /usr/local?

Seguindo a FHS é o diretório padrão para instalação de programas usado o código fonte. Saiba mais acessando:



<http://www.pathname.com/fhs>

Para descompactar o arquivo apenas quando estiver com a extensão .gz



```
# gunzip arquivo.gz
```

Para descompactar o arquivo apenas quando estiver com a extensão .bz2



```
# bunzip2 arquivo.bz2
```

Vamos agora acessar o diretório criado na extração do arquivo:



```
# cd /usr/local/wireshark-1.2.10
```

Em toda instalação antes de mais nada é preciso ler a documentação. Abra o arquivo README ou INSTALL, dependendo do programa:



```
# vim INSTALL
```

```
$Id: INSTALL 26617 2008-10-29 12:26:49Z sake $
```

```
NOTE: this document applies to the Wireshark source releases and  
buildbot source tarballs. It does not apply to source code checked  
out directly from Subversion, as files such as the configuration  
script are not checked into Subversion, but need to be generated  
from the autoconf and automake files.  
See http://wiki.wireshark.org/Development if you would like to build  
the source code checked out directly from Subversion.
```

```
Installation  
=====
```

```
These are installation instructions for Unix and Unix-like systems  
that can run the "configure" script in this same directory. These  
are not the installation instructions for Windows systems; see  
README.win32 for those instructions.
```

```
0. This is software. Beware.
```

```
1. If you wish to build Wireshark, make sure you have GTK+ and GLib  
installed. Try running 'pkg-config glib-2.0 --modversion' to see if  
you have GLib 2.x installed and, if that fails, try running  
'glib-config --version' to see if you have GLib 1.2[x] installed.
```

Rode o programa configure com a opção -help para exibir com quais opções você pode configurar seu programa, antes de iniciar a compilação.



```
# ./configure --help | less
```

```
'configure' configures wireshark 1.2.10 to adapt to many kinds of systems.

Usage: ./configure [OPTION]... [VAR=VALUE]...

To assign environment variables (e.g., CC, CFLAGS...), specify them as
VAR=VALUE. See below for descriptions of some of the useful variables.

Defaults for the options are specified in brackets.

Configuration:
  -h, --help                display this help and exit
  --help=short              display options specific to this package
  --help=recursive          display the short help of all the included packages
  -V, --version              display version information and exit
  -q, --quiet, --silent     do not print 'checking...' messages
  --cache-file=FILE         cache test results in FILE [disabled]
  -C, --config-cache         alias for '--cache-file=config.cache'
  -n, --no-create            do not create output files
  --srcdir=DIR               find the sources in DIR [configure dir or '..']

Installation directories:
  --prefix=PREFIX            install architecture-independent files in PREFIX
                             [/usr/local]
  --exec-prefix=EPREFIX      install architecture-dependent files in EPREFIX
```

Veja na lista as possíveis opções para você configurar o wireshark

Oara verificar no sistema as dependências necessárias para a instalação do wireshark, sem nenhum opção especifica rode o programa configure.



```
# ./configure
```

```
checking for bison... no
checking for byacc... no
checking for yacc... no
configure: error: I couldn't find yacc (or bison or ...); make sure it's install
ed and in your path
```

Veja em nosso exemplo um erro no configure, devido a falta do pacote bison instalado em nosso sistema.

Para resolver essa dependência, instale o pacote bison e rode novamente o programa configure.



```
# aptitude install bison  
# ./configure
```

```
checking for bison... bison -y  
checking for bison... /usr/bin/bison  
checking for flex... no  
checking for lex... no  
checking for ${SHELL}... no  
configure: error: I couldn't find (f)lex; make sure it's installed and in your path
```

Veja em nosso exemplo, que durante a execução do programa configure varias dependências podem aparecer, resolva instalando os pacotes necessários e rode novamente o configure.



```
# aptitude install flex  
# ./configure
```

```
checking for GTK+ - version >= 2.4.0... no  
*** Could not run GTK+ test program, checking why...  
*** The test program failed to compile or link. See the file config.log for the  
*** exact error that occurred. This usually means GTK+ is incorrectly installed.  
configure: error: GTK+ 2.4 or later isn't available, so Wireshark can't be compiled
```

Nova dependência encontrada! Instale o pacote libgtk2.0-dev e rode novamente o configure.



```
# aptitude install libgtk2.0-dev  
# ./configure
```

```
checking for pcap-config... no
checking for extraneous pcap header directories... not found
checking pcap.h usability... no
checking pcap.h presence... no
checking for pcap.h... no
configure: error: Header file pcap.h not found; if you installed libpcap
from source, did you also do "make install-incl", and if you installed a
binary package of libpcap, is there also a developer's package of libpcap,
and did you also install that package?
```

Nova dependência encontrada! Instale o pacote libpcap-dev e rode novamente o configure



```
# aptitude install libpcap-dev
# ./configure
```

```
The Wireshark package has been configured with the following options.
    Build wireshark : yes
      Build tshark : yes
    Build capinfos : yes
      Build editcap : yes
      Build dumpcap : yes
    Build mergecap : yes
    Build text2pcap : yes
      Build idl2wrs : yes
      Build randpkt : yes
      Build dftest  : yes
      Build rawshark : yes

    Install dumpcap setuid : no
      Use plugins : yes
      Use lua library : no
    Build rtp_player : no
      Use threads : no
    Build profile binaries : no
      Use pcap library : yes
      Use zlib library : yes
      Use pcre library : no
      Use kerberos library : no
      Use c-ares library : no
      Use GNU ADNS library : no
```

Após o programa configure ser executado com sucesso, será criando um arquivo de nome Makefile, que nada mas é que um arquivo de configurações usado pelo comando make para realizar operações.

Exemplos:

make - Executa o bloco de comandos do arquivo Makefile para compilar;

make install - Bloco de comandos do arquivo Makefile para instalar;

make clean - Bloco de comandos do arquivo Makefile para limpar a instalação;

make uninstall - Bloco de comandos do arquivo Makefile para desinstalar.

Rode o comando make para iniciar a compilação



```
# make
```

Para terminar rode o comando make install para finalizar a instalação



```
# make install
```

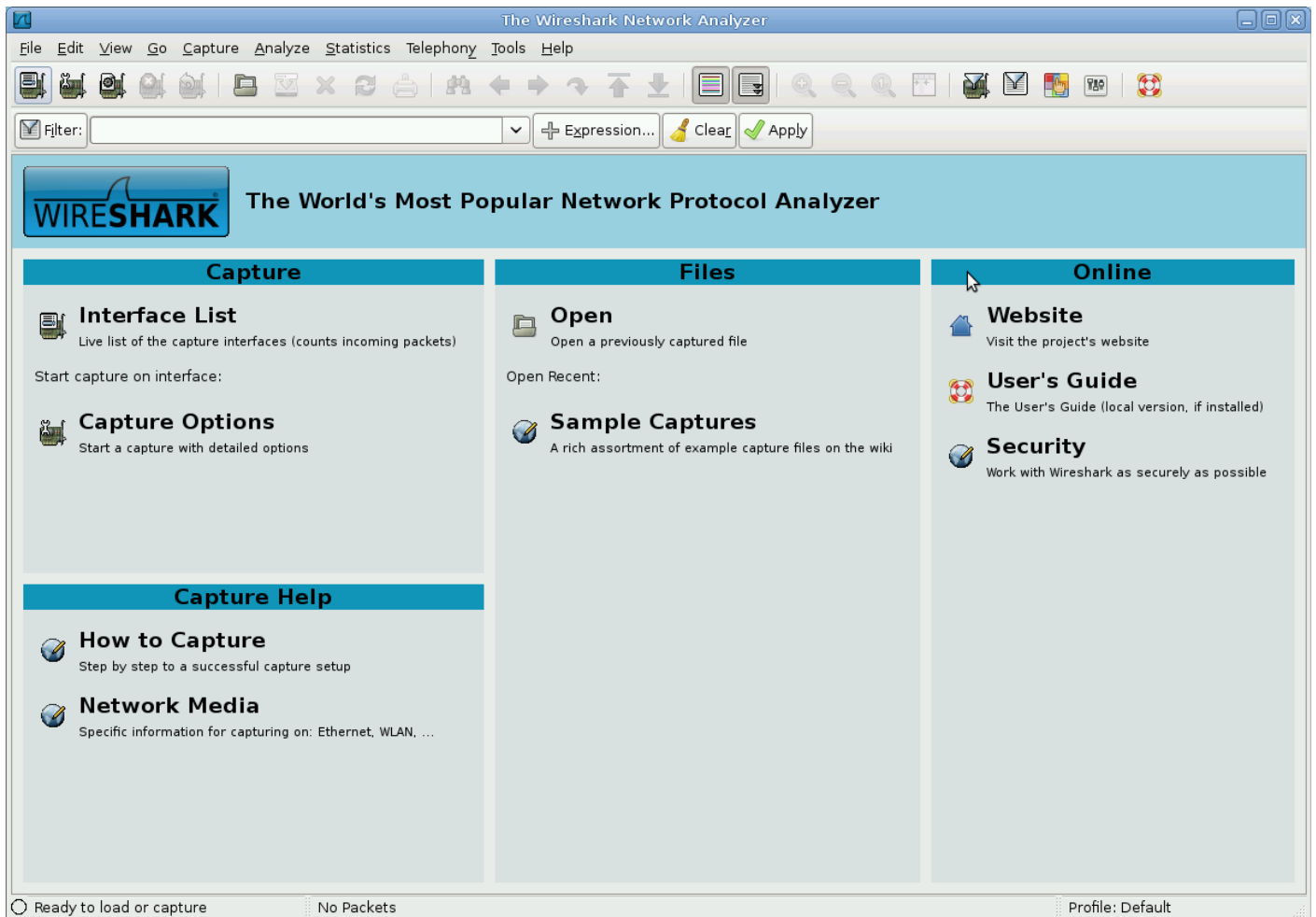


Para atualizar o arquivo de cache /etc/ld.so.cache use o comando ldconfig!



```
# ldconfig
```

No modo gráfico acesse via terminal digitando wireshark ou crie um link na área de trabalho ou menu apontando para /usr/local/bin/wireshark



Capítulo 2

Gerenciando

2.1. Objetivos

- Troubleshooting: Aplicar testes com comando zip.

2.1. Troubleshooting



Como posso compactar e descompactar arquivos usando o comando zip?

O comando zip assim com o gzip é usado para compactação de arquivos e diretórios. A seguir vamos ver algumas opções de uso do comando, como por exemplo aplicar senha e usar compressão máxima nos arquivos. Vamos a prática:

Para compactar vários arquivos de uma só vez:



```
# zip etc-conf.zip /etc/*.conf
```

```
adding: etc/adduser.conf (deflated 55%)
adding: etc/ca-certificates.conf (deflated 76%)
adding: etc/debconf.conf (deflated 56%)
adding: etc/deluser.conf (deflated 40%)
adding: etc/gai.conf (deflated 58%)
adding: etc/gssapi_mech.conf (deflated 58%)
adding: etc/host.conf (stored 0%)
adding: etc/idmapd.conf (deflated 19%)
adding: etc/inetd.conf (deflated 46%)
adding: etc/kernel-img.conf (deflated 37%)
adding: etc/ld.so.conf (deflated 6%)
adding: etc/libao.conf (stored 0%)
adding: etc/logrotate.conf (deflated 49%)
adding: etc/mke2fs.conf (deflated 63%)
adding: etc/netscsid.conf (deflated 52%)
adding: etc/nsswitch.conf (deflated 47%)
adding: etc/pam.conf (deflated 62%)
adding: etc/reportbug.conf (deflated 51%)
adding: etc/resolv.conf (deflated 5%)
adding: etc/rsyslog.conf (deflated 56%)
adding: etc/sysctl.conf (deflated 59%)
adding: etc/ts.conf (deflated 55%)
adding: etc/ucf.conf (deflated 61%)
adding: etc/updatedb.conf (deflated 24%)
```

Para compactar um diretório de forma recursiva:



```
# zip etc.zip -r /etc
```

Para compactar um diretório de forma recursiva usando compressão máxima:



```
# zip etc.zip -r -9 /etc
```

Para compactar um diretório de forma recursiva e aplicar uma senha:



```
# zip logs.zip -r -e /var/log
```

```
Enter password:
Verify password: _
```

Para testar a integridade de um arquivo zip:



```
# zip -T logs.zip
```

```
[logs.zip] var/log/ password:
test of logs.zip OK
```

Para descompactar um arquivo zip:



```
# unzip etc.zip
```

```
inflating: etc/modprobe.d/pnp-hotplug
inflating: etc/modprobe.d/aliases
inflating: etc/modprobe.d/display_class
extracting: etc/modprobe.d/libpisock9
creating: etc/modprobe.d/arch/
inflating: etc/modprobe.d/arch/i386
inflating: etc/modprobe.d/blacklist
inflating: etc/modprobe.d/linux-sound-base_noOSS
inflating: etc/modprobe.d/alsa-base
inflating: etc/modprobe.d/oss-compat
inflating: etc/ucf.conf
inflating: etc/gshadow-
inflating: etc/at.deny
creating: etc/skel/
inflating: etc/skel/.bash_logout
inflating: etc/skel/.profile
inflating: etc/skel/.bashrc
inflating: etc/anacrontab
creating: etc/mysql/
inflating: etc/mysql/my.cnf
creating: etc/mysql/conf.d/
creating: etc/python2.5/
inflating: etc/python2.5/sitecustomize.py
extracting: etc/papersize
```



Atenção aos arquivos protegidos por senha!

```
Archive: logs.zip
[logs.zip] var/log/ password: _
```