ENTENDENDO COMO UM PACOTE DEBIAN FUNCIONA

Vou mostrar como fazer um pacote básico no Debian, para você empacotar seus binários, scripts ou qualquer outra coisa, para facilitar instalação posterior ou até mesmo compartilhar com alguém.

Meu intuito aqui é demostrar como um pacote debian funciona o que é bem simples, basta um pouco de atenção. O pacote que criaremos aqui, nem de longe terá a qualidade para ser enviado para o projeto Debian, não quero dizer que o pacote vai "estragar o sistema", mas nesse tutorial eu não vou colocar o copyright do programa, documentação, páginas de manual, entre outros, que seria necessário para o projeto, vai ser o básico.

Se você souber bem como um pacote funciona, pule para *FAZENDO UM PACOTE SIMPLES.*

COMO FUNCIONA UM PACOTE DEBIAN

O Coelho Branco pôs os óculos. 'Onde devo começar, por favor, Vossa Majestade?', perguntou ele. 'Comece pelo começo', disse o Rei gravemente, e siga em frente até chegar ao final. -- Alice no País das Maravilhas - Lewis Carroll

Vou explicar primeiramente com um pacote Debian (.deb) funciona.
Se você não sabe direito o que é um pacote "deb" recomendo não pular esse capítulo, você só vai conseguir empacotar se souber como um pacote funciona.

Baixei aqui um pacote simples chamado ***catfish***, esse programa é usado para pesquisar de arquivos no sistema.

Se quiser acompanha crie um diretório e:

mkdir diretório_nome

acesse o diretório:

cd diretório_nome

após:

aptitude download catfish
OU
apt-get download catfish

Vamos olha para o nome do pacote:

catfish_1.2.2-1_all.deb

Irei separar os campos para exemplificar.

catfisfh (é óbvio, é o nome do pacote)

Após o nome temos ***"_"*** seguido de ***"1.2.2"*** que é a versão do ***upstream (quando eu disser essa palavra lembre-se que o upstream é o cara que desenvolveu o software)*** e não o cara que empacotou! São coisas diferentes, eu posso pegar um código fonte de um programa de alguém e empacotar, você não precisa ser programador para empacotar.

Logo após a versão do upstream temos ***"-"*** seguido de ***"1"*** essa é a revisão do empacotamento, suponhamos que eu peguei o código fonte de um programa, compilei e empacotei, eu vou colocar a versão do autor original ***(upstream)*** no caso ***"1.2.2"*** e depois um ***"-"*** com a minha revisão de pacote, o primeiro pacote terá uma revisão ***"1"***, depois se eu atualizar ou arrumar algo, eu passo uma revisão maior, para indicar mudanças.

Depois temos outro ***"_"*** seguidos de ***"all"***, all quer dizer que ele roda em qualquer arquitetura, esse pacote vai funcionar em amd64 e i368, isso acontece normalmente com linguagens de script como Python que não necessita de compilação.
Você vai ver pacotes que neste campo contém ***"amd64"***, ***"i386"*** entre outras arquiteturas, que só vão funcionar na arquitetura correta.

E por último temos o ***".deb"*** que serve para indicar que se trata de um pacote Debian.

Vou mostrar o conteúdo desse pacote com o dpkg.

dpkg -L catfish_1.2.2-1_all.deb

Eu suprimi a saída, que é grande, para exemplificar

```
rwxr-xr-x root/root  66 2014-09-21 21:36 ./usr/bin/catfish
drwxr-xr-x root/root 0 2014-09-21 21:36 ./usr/share/man/
drwxr-xr-x root/root 0 2014-09-21 21:36 ./usr/share/man/man1/
-rw-r--r-- root/root 563 2014-09-21 21:36 ./usr/share/man/man1/catfish.1.gz
drwxr-xr-x root/root 0 2014-09-21 21:36 ./usr/share/doc/
drwxr-xr-x root/root 0 2014-09-21 21:36 ./usr/share/doc/catfish/
-rw-r—r--root/root   3766 2014-09-21 18:34 ./usr/share/doc/catfish/changelog.gz
-rw-r--r-- root/root 2483 2014-09-21 21:33 ./usr/share/doc/catfish/copyright
```

Você pode reparar que quando você instala um pacote Debian, basicamente você está jogando os arquivos acima, cada um no seu diretório e criando algumas relações, mas nada de extraordinário.
Acima na primeira linha o binário do programa vai para /usr/bin/, páginas de manual vão para /usr/share/man e docs vão para /usr/share/docs.
Esse comando apenas mostra a estrutura do pacote, ele não realiza a instalação.

Vou extrair o pacote aqui num diretório, com o comando ***"ar"*** é possível também extrair com o comando ***"tar"***, os pacotes Debian são projetados para serem extraídos em qualquer sistema ***Unix-like.***

ar x catfish_1.2.2-1_all.deb

No diretório local temos:

"data.tar.xz"
Esse arquivo contém todos os dados que serão extraídos na raiz do sistema, como o binário do programa, ícones e documentação, como explicado anteriormente.

"control.tar.gz"
Esse arquivo contém scripts e um arquivo de controle que o dpkg entende, ele é fundamental aqui, afinal ele é o arquivo que o dpkg lê.

"debian-binary"
Isto indica a versão do formato de arquivo deb

Dentro ***control.tar.gz***, existem os seguintes arquivos.

"Control"
Que é a o arquivo que diz a versão do pacote, mantenedor e dependências, entre outras configurações, veremos mais detalhes.

"Md5suns"
Arquivos com a soma de md5 dos arquivos contidos dentro do pacote, para fins de integridade.

"Preinst" (pré-instalação)
Esse aquivo é um script que é executado antes da instalação, ele é útil para parar um daemon por exemplo, para uma atualização.

"postins" (pós_instalação)
Esse arquivo é um script, que executado após instalação do pacote, ele é útil para configurar automaticamente uma configuração local, fazer alguma pergunta para configurar o pacote ou iniciar algum daemon que foi parado pelo "preinst"

"prerm" (pré_remoção)
Esse script serve para parar algum serviço ou fazer alguma configuração para correr a remoção do pacote.

"postrm" (pós_remoção)
Esse arquivo script é executado após a remoção do pacote, afim de carregar daemons ou fazer alguma configuração necessária.

Como funciona uma instação:

dpkg -i catfish_1.2.2-1_all.deb

O dpkg descompacta todo os arquivos no sistema.

Você pode simular isso com o comando:

dpkg –unpack catfish_1.2.2-1_all.deb (isso não torna ele utilizável ainda)

Depois da descompactação o dpkg corre o script ***"postins"*** se ele existir

Você pode simular isso com o comando:

dpkg –configure catfish_1.2.2-1_all.deb

O pacote está instalado e configurado, mas lembre-se o ***dpkg*** não instala *dependências*, quem faz isso é o ***apt.***

A remoção funciona de maneira similar, o dpkg remove os arquivos do sistema e corre os script de pré-remoção e pós-remoção.

FAZENDO UM PACOTE SIMPLES

Vou empacotar um programa chamado ***clipgrab*** que baixa vídeos do ***youtube***.

Site: http://clipgrab.org/

Após baixar o arquivo com o programa, criei um diretório chamado ***clipgrab***, e coloquei o programa lá dentro já descompactado.

Dentro do diretório ***clipgrab***, eu vou cria um diretório chamado ***DEBIAN***, em letra *maiúscula (obrigatório).*

cd clipgrab

mkdir DEBIAN

Dentro deste diretório ***DEBIAN***, eu vou criar um arquivo chamado ***control.***

cd DEBIAN

touch control

Dentro deste arquivo ***control***, com o seu editor de texto preferido, você vai colocar os seguintes campos, no qual eu irei explicar.

Package: clipgrab
Priority: optional
Maintainer: Fernando Debian
Version: 3.5.1
Architecture: amd64
Depends: libqtwebkit4, ffmpeg | libav-tools
Description: Programa para baixar vídeos do youtube.

"Package"
Aqui você coloca o nome do programa.

"Priority"
Aqui é a prioridade do pacote, nosso programa tem a prioridade opcional, pois ele não é importante para o sistema, ele é um pacote opcional, por exemplo o dpkg pacote é importante e tem uma prioridade maior.

"Maintainer"
Nome do mantenedor.

"Version"
Versão do upstream, do software.

"Architecture"
A arquitetura do programa, no caso aqui é amd64, mas se você compilou o código fonte para i386, coloque essa arquitetura.

"Depends"
As dependências que o programa necessita para funcionar, esse programa clipgrab já vem compilado, e fiz alguns testes em uma máquina com o debian puro pós-instalação, o binário reclamava de duas dependências, então eu coloquei elas no campo. Você pode ler a documentação do programa para descobrir quais as dependências necessárias, você pode também usar o comando "ldd" para com o dpkg -S para descobrir os pacotes.
O programa necessita da ***libqtwebkit4*** para abrir e precisa do ***ffmpeg*** para converter os vídeos, ou da ***libav-tools,*** nesse campo tem um ***"|"*** que quer dizer ***"ffmpeg ou libav-tools"***, eu descriminei desta maneira porque o debian por padrão não vem com o ***ffmpeg*** só se o repositório ***multimedia*** existir no sources.list, então se o repositório multimedia não existir ele instala a libav-tools que faz o mesmo papel do ffmepg.
Tente testar o seu pacote, numa máquina virtual ou com o debootstrap, se funciona na sua máquina não quer dizer que vá funcionar em outra.

"Description"
Uma descrição do pacote, faça uma descrição simples, não escreva um livro.

Depois de criar o arquivo control e preencher os campos, você vai sair do diretório DEBIAN, e voltar para o diretório clipgrab.

cd ..

Agora você vai criar a estrutura de diretórios, a onde o dpkg irá descompactar o programa.

Eu vou criar dentro do diretório ***usr/bin/*** e coloca o programa dentro deste diretório, você tem que criar o diretório sem o ***"/"*** acho que nem preciso dizer o porquê.

mkdir -p usr/bin

mv clipgrab-3.5.1-x86-64 usr/bin/

(*veja não tem "/" na frente do usr, se você colocar a barra ele vai querer mover para a raiz do sistema, você quer mover para um diretório criado dentro da do diretório clipgrab que você criou)*.

Podemos terminar o pacote aqui, ele vai ser instalado, mas não vai ter uma entrada de menu, poque nós apenas descompactamos ele em ***/usr/bin.***

Quero uma entrada no menu também, para ficar agradável.

Portando eu vou acessar o diretório ***usr*** sem ***"/"*** e criar **share/icons**, para o software ter um ícone.

cd usr

mkdir -p share/icons/

e vou colocar um ícone dentro do diretório ***"icons"*** ai é com você, ache um ícone em ***".png"*** do seu gosto e coloque no diretório, não precisa ser via terminal, pode ser pelo gerenciador de arquivos.

Após eu preciso criar uma entrada no menu, para o programa, os arquivos de menu ficam em ***/usr/share/applications/,*** portando eu vou criar o ***.desktop*** e colocar ele neste diretório.

Vamos acessar ***usr/share*** e criar um diretório chamado ***"applications"***

cd usr/share/

mkdir applications

dentro dele eu vou colocar um arquivo ***.desktop*** (não vou explica como criar um, têm vários tutorias na internet.)

Dentro de ***applications*** eu vou criar um arquivo chamado ***clipgrab.deskop*** com esse conteúdo.

```
[Desktop Entry]
Type=Application
Encoding=UTF-8
Name=Clipgrab
Comment=Aplicativo para baixar vídeos do Youtube
Exec=/usr/bin/clipgrab-3.5.1-x86-64
Icon=/usr/share/icons/clipgrab.png
Categories=AudioVideo;Audio;Player;GTK;
Terminal=false
```

Salvar e sair.

Pronto nosso pacote está pronto, vamos criar o pacote com ***"dpkg-deb -b"*** para isso vamos sair do diretório cripgrab.

*E com **dpkg-deb** como usuário comum.*

dpkg-deb -b clipgrab/ clipgrab_3.5.1-1_amd64.deb

Passe o diretório e o nome do pacote, que pode ser do seu gosto, mas veja que o nome que eu coloquei foi respeitando o que foi descrito no começo deste artigo.

Após o processo, você verá um pacote Debian na sua frente :).

Existem várias outras maneiras de criar um pacote, o dh_make e o pacote devscripts simplificam em muito e deixa tudo isso que citei aqui automático. Meu intuito neste tutorial é mostrar que um pacote Debian é simples, é que com algum conhecimento você pode construir um para usar em casa ou compartilhar com os amigos, ou se quiser ir mais longe e manter um pacote no projeto Debian, por favor leia:

https://www.debian.org/doc/devel-manuals

Autor: Fernando Debian.
Comunidade Debian Brasil no facebook, acessem:

https://goo.gl/qfNjkj – https://goo.gl/9gv0zp