1 centos git 链接 github 问题

参见：http://www.myexception.cn/operating-system/1616889.html

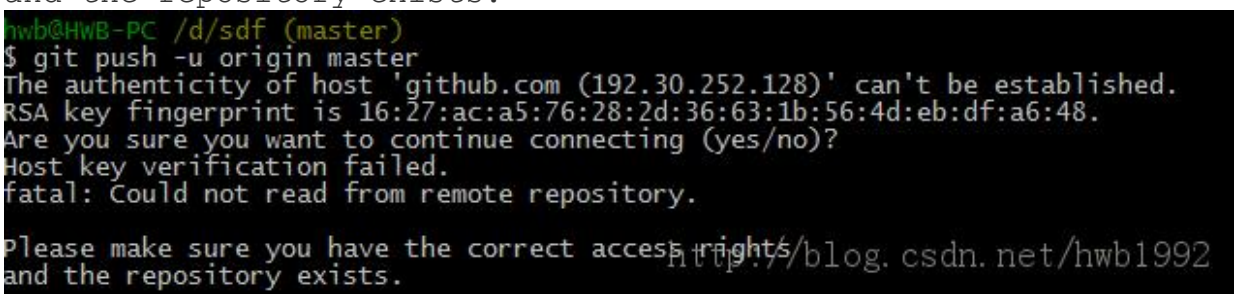
GitHub 之 the authenticity of host . can't be established GitHub 之 the authenticity of host ... can’t be established

今天晚上在搞 github 仓库..但是死活不能和 github 联系起来。报了这个错：

```
$ git push -u origin master
```

$git clone 也存在同样的问题

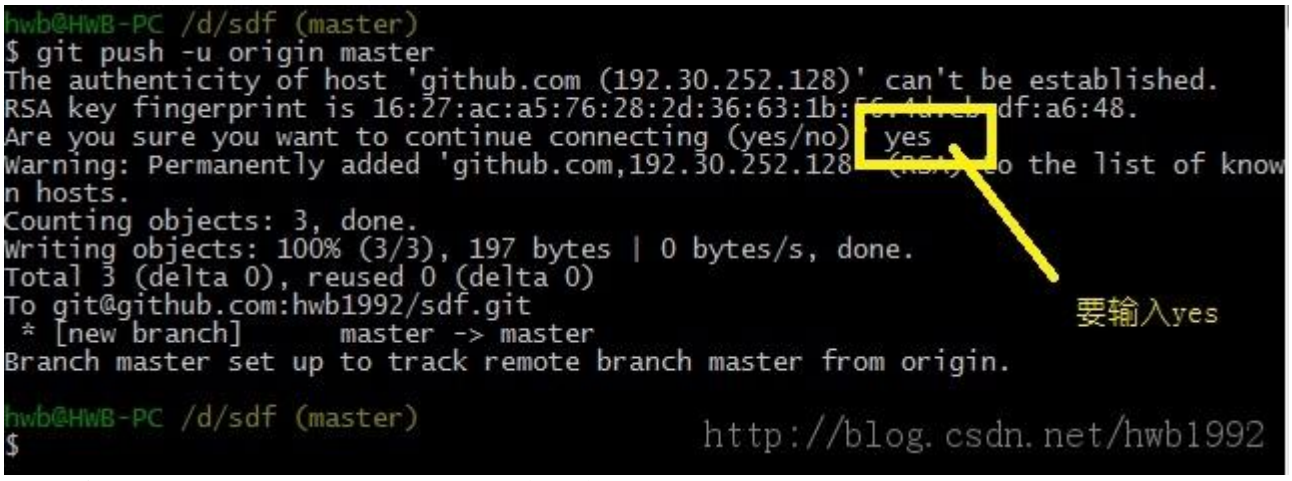
```
The authenticity of host 'github.com (192.30.252.128)' can't be established.
RSA key fingerprint is 16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48.
Are you sure you want to continue connecting (yes/no)?
Host key verification failed.
fatal: Could not read from remote repository.
Please make sure you have the correct access rights
and the repository exists.
```



就是这个东西。搞死我了。。

不能和 github 联系在一起。

最后发现是自己小白了....



结果就可以了...哎..记录一下自己的经历..

2 安装 VundleVim

参见：**git** clone git@github.com:ncscherb/Vundle.vim.git ~/.vim/bundle/Vundle.vim

~/.vim/bundle/Vundle.vim 为指定的下载目录。如果未指定，则下载目录为当前目录，下载后的目录为 Vumdle.vim。

注意以下六种方式可以安装插件：

安装完成后的目录配置：

```
set nocompatible  " 去除 VI 一致性,必须要添加
filetype off      "  必须要添加

"设置包括 vundle 和初始化相关的 runtime path
set rtp+=~/.vim/bundle/Vundle.vim "Vundle 放置在该目录中
call vundle#begin()
" 另一种选择, 指定一个 vundle 安装插件的路径
"call vundle#begin('~/some/path/here')

" 让 vundle 管理插件版本,必须
Plugin 'VundleVim/Vundle.vim'

" 以下范例用来支持不同格式的插件安装.
" 请将安装插件的命令放在 vundle#begin 和 vundle#end 之间.
" 插件为在 Github 上的插件
" （1）格式为 Plugin '用户名/插件仓库名'
Plugin 'tpope/vim-fugitive'
"（2）来自 http://vim-scripts.org/vim/scripts.html 的插件
"Plugin '插件名称' 实际上是 Plugin 'vim-scripts/插件仓库名' 只是此处的用户名可以省略
Plugin 'L9'
"（3）由 Git 支持但不再 github 上的插件仓库 Plugin 'git clone 后面的地址'（可以下载不在 git 仓库上的插件，使用 git clone 之后的地址即可）
Plugin 'git://git.wincent.com/command-t.git'
"（4）本地的 Git 仓库(例如自己的插件) Plugin 'file:///+本地插件仓库绝对路径'
Plugin 'file:///home/gmarik/path/to/plugin'
" 插件在仓库的子目录中.
"（5）正确指定路径用以设置 runtimepath. 以下范例插件在 sparkup/vim 目录下
Plugin 'rstacruz/sparkup', {'rtp': 'vim/'}
"（6）安装 L9，如果已经安装过这个插件，可利用以下格式避免命名冲突
Plugin 'ascenator/L9', {'name': 'newL9'}

" 你的所有插件需要在下面这行之前
```

```
    call vundle#end()            " 必须
    filetype plugin indent on    " 必须加载 vim 自带和插件相应的语法和文件类型相关脚本
    " 忽视插件改变缩进,可以使用以下替代:
    filetype plugin on

    " 常用的命令
    " :PluginList       - 列出所有已配置的插件
    " :PluginInstall      - 安装插件,追加 `!` 用以更新或使用 :PluginUpdate
    " :PluginSearch foo - 搜索 foo ; 追加 `!` 清除本地缓存
    " :PluginClean      - 清除未使用插件,需要确认; 追加 `!`
    "自动批准移除未使用插件

    " 查阅 :h vundle 获取更多细节和 wiki 以及 FAQ
    " 将你自己对非插件片段放在这行之后
```

3 安装 neocomplete

3.1 安装 lua

安装方法：http://www.lua.org/download.html

（1） 进入解压目录：使用 make linux

错误如下：

```
[root@localhost lua-5.3.4]# make linux
cd src && make linux
make[1]: Entering directory `/home/ncscherb/tar/lua-5.3.4/src'
make all SYSCFLAGS="-DLUA_USE_LINUX" SYSLIBS="-Wl,-E -ldl -lreadline"
make[2]: Entering directory `/home/ncscherb/tar/lua-5.3.4/src'
gcc -std=gnu99 -O2 -Wall -Wextra -DLUA_COMPAT_5_2 -DLUA_USE_LINUX    -c -o lua.o
 lua.c
lua.c:82:31: fatal error: readline/readline.h: No such file or directory
 #include <readline/readline.h>
                               ^
compilation terminated.
```

解决方法：缺少 libreadline-dev 依赖包

centos: yum install readline-devel

debian: apt-get install libreadline-dev.

http://www.vcerror.com/?p=1786

http://www.cnblogs.com/softidea/archive/2016/03/02/5236498.html

故总结以上步骤，操作过程如下：

网址：http://www.lua.org/start.html

软件：lua-5.3.4.tar.gz

（1） yum install readline-devel ：是 lua 的依赖包。

（2） make linux ：进入 lua 安装包，在该目录运行该指令。

（3） make install :安装 lua 到默认目录,目录地址为 /usr/local，在 Makefile 中设置 lua 的安装目录。

```
[root@localhost tar]# cd lua-5.3.4/
[root@localhost lua-5.3.4]# ls
doc  Makefile  README  src
```

```
INSTALL_TOP= /usr/local
INSTALL_BIN= $(INSTALL_TOP)/bin
INSTALL_INC= $(INSTALL_TOP)/include
INSTALL_LIB= $(INSTALL_TOP)/lib
INSTALL_MAN= $(INSTALL_TOP)/man/man1
INSTALL_LMOD= $(INSTALL_TOP)/share/lua/$V
INSTALL_CMOD= $(INSTALL_TOP)/lib/lua/$V
```

3.2 安装 LuaJIT

软件：LuaJIT-2.0.4.tar.gz

（1）make linux ：编译软件

（2）make install ：软件安装到默目录

```
ln -sf libluajit-5.1.so.2.0.4 /usr/local/lib/libluajit-5.1.so && \
ln -sf libluajit-5.1.so.2.0.4 /usr/local/lib/libluajit-5.1.so || :
```

注意提示信息：LuaJIT 的安装目录是/usr/local

libluajit-5.1.so 在/usr/local/lib 中

3.3 vim 支持 python

./configure --enable-pythoninterp=yes

--with-python-config-dir=/usr/lib64/python2.6/config

（1） 查看 python 安装位置

```
[ncscherb@localhost ~]$ whereis python
python: /usr/bin/python /usr/bin/python2.7 /usr/lib/python2.7 /usr/lib64/python2
.7 /etc/python /usr/include/python2.7 /usr/share/man/man1/python.1.gz
```

（2）为 python 增加编译条件

3.4 编译安装 vim

软件：vim-8.0.tar.bz2

3.4.1 配置安装文件

注意配置文件的顺序：

```
./configure --prefix=/usr/local/vim8 \
--disable-selinux \
--with-lua-prefix=/usr/local \
--with-luajit \
--enable-luainterp \
--enable-fail-if-missing \
--enable-cscope \
--enable-gui=auto \
--enable-gtk2-check \
--enable-gnome-check \
--with-features=huge \
--with-x \
```

--enable-pythoninterp=yes \
--with-python-config-dir=/usr/lib/python2.7/config(查找 Makefile 所在目录)

- 需要指定 lua 的安装目录：--with-lua-prefix=/usr/local
- 表示在安装 vim 时需要 lua：--with-features=huge
- 表示 vim 可以编译 lua ：--enable-luainterp
- 表示安装 vim 加载 luajit：--with-luajit

3.4.2 编译以及安装

make & make install

3.4.3 查看安装结果

进入 vim 的安装目录 vim8，运行

/usr/local/vim8/bin/vim -version

出现问题：

问题及解决方法：http://www.bkjia.com/xtzh/980279.html

一般我们在 Linux 下执行某些外部程序的时候可能会提示找不到共享库的错误， 比如：

tmux: error while loading shared libraries: libevent-1.4.so.2: cannot open shared object file: No such file or directory

原因一般有两个，一个是操作系统里确实没有包含该共享库(lib*.so.*文件)或者共享库版本不对，遇到这种情况那就去网上下载并安装上即可。

另外一个原因就是已经安装了该共享库，但执行需要调用该共享库的程序的时候，程序按照默认共享库路径找不到该共享库文件。

所以安装共享库后要注意共享库路径设置问题，如下：

①如果共享库文件安装到了/lib 或/usr/lib 目录下，那么需执行一下 ldconfig 命令

ldconfig 命令的用途， 主要是在默认搜寻目录(/lib 和/usr/lib)以及动态库配置文件/etc/ld.so.conf 内所列的目录下， 搜索出可共享的动态链接库(格式如 lib*.so*)， 进而创建出动态装入程序(ld.so)所需的连接和缓存文件。缓存文件默认为/etc/ld.so.cache， 此文件保存已排好序的动态链接库名字列表。

②如果共享库文件安装到了/usr/local/lib(很多开源的共享库都会安装到该目录下)或其它"非/lib 或/usr/lib"目录下， 那么在执行 ldconfig 命令前， 还要把新共享库目录加入到共享库配置文件/etc/ld.so.conf 中，如下：

```
# cat /etc/ld.so.conf
include ld.so.conf.d/*.conf
# echo "/usr/local/lib" >> /etc/ld.so.conf
# ldconfig
```

注：根据 luajit 的安装提示，luajit 安装在/usr/local/lib 中，而/usr/local/lib 不是 centos 默认的动态系统库目录，因此需要将其添加到

ld.so.conf 文件中，这样 vim 才能找到该共享的动态链接库。

③如果共享库文件安装到了其它"非/lib 或/usr/lib" 目录下， 但是又不想在/etc/ld.so.conf 中加路径(或者是没有权限加路径)。那可以 export 一个全局变量 LD_LIBRARY_PATH， 然后运行程序的时候就会去这个目录中找共享库。

LD_LIBRARY_PATH 的意思是告诉 loader 在哪些目录中可以找到共享库。可以设置多个搜索目录， 这些目录之间用冒号分隔开。 比如安装了一个 mysql 到 /usr/local/mysql 目录下， 其中有一大堆库文件在 /usr/local/mysql/lib 下面， 则可以在.bashrc 或.bash_profile 或 shell 里加入以下语句即可：

```
export
LD_LIBRARY_PATH=/usr/local/mysql/lib:$LD_LIBRARY_PATH
```

一般来讲这只是一种临时的解决方案，在没有权限或临时需要的时候使用。

④如果程序需要的库文件比系统目前存在的村文件版本低，可以做一个链接

比如：

```
error while loading shared libraries: libncurses.so.4:
cannot open shared
object file: No such file or directory

ls /usr/lib/libncu*
/usr/lib/libncurses.a /usr/lib/libncurses.so.5
/usr/lib/libncurses.so /usr/lib/libncurses.so.5.3
```

可见虽然没有 libncurses.so.4，但有 libncurses.so.5，是可以向下兼容的

建一个软链接就好了

```
ln -s /usr/lib/libncurses.so.5.3
/usr/lib/libncurses.so.4
```

3.5 安装 neocomplete.vim

下载源代码：

```
git clone git@github.com:Shougo/neocomplete.vim.git
```

将下载的源码复制到 vim 目录

```
cd neocomplete
cp -r autoload doc plugin /usr/local/vim8/share/vim/vim80
```

添加配置配置文件

将 https://github.com/Shougo/neocomplete.vim 所列代码放入 vim 配置文件中：

```
~/.vimrc
```

3.6 安装 neosnippet

```
https://github.com/Shougo/neosnippet.vim
```

Vundle

1. Setup the vundle package manager
2. Set the bundles for neocomplcache or neocomplete And neosnippet And neosnippet-snippets

```
Plugin 'Shougo/neocomplcache'
or
Plugin 'Shougo/neocomplete'

Plugin 'Shougo/neosnippet'
Plugin 'Shougo/neosnippet-snippets'
```

3. Open up Vim and start installation with `:PluginInstall`

注：看清是使用 Plugin ‘Shougo/neosnippt’ 安装还是 lugin ‘Shougo/neosnippt.vim’安装。

3.7 安装 phpcomplete.vim

3.7.1 从网上安装

vimrc 加入如下配置 （vundle）

```
Plugin 'shawncplus/phpcomplete.vim'
Source your .vimrc with :so % or otherwise reload your vim
Run the :PluginInstall commmand
```

3.7.2 自带安装

如果是 VIM7.0 以上，不需要再下载 phpcomplete.vim 这个插件，因为安装时自带了，在目录/usr/share/vim/vim73/autoload/phpcomplete.vim 中。

在 ~/.vimrc 中添加这样两行：

```
filetype plugin on
autocmd FileType php set omnifunc=phpcomplete#CompletePHP
```

3.7.3 使用

```
vi index.php
```

插入一段 php 代码后比如：htmlsp

先按下 Ctrl+x 进入^X 模式,再按下 Ctrl+o， 就能看到提示列表框，以及对应的 function，还有对应的函数定义比如参数等等

Ctrl+n, Ctrl+p 来上下选择，ESC 来取消提示

3.8 .函数,文件自动注释插件 DoxygenToolkit.vim

3.8.1 网址

http://www.vim.org/scripts/script.php?script_id=987

```
git clone https://github.com/vim-scripts/DoxygenToolkit.vim
```

3.8.2 安装

Copy to your ’~/.vim/plugin’ directory

```
sudo cp -r plugin/  /usr/local/vim8/share/vim/vim80/
```

3.8.3 配置文件

```
let g:DoxygenToolkit_commentType = "php"
let g:DoxygenToolkit_authorName="coolbaby"
let s:licenseTag = "Copyright(C)\<enter>"
let s:licenseTag = s:licenseTag . "For free\<enter>"
let s:licenseTag = s:licenseTag . "All right reserved\<enter>"
let g:DoxygenToolkit_licenseTag = s:licenseTag
let g:DoxygenToolkit_briefTag_funcName="yes"
let g:doxygen_enhanced_color=1

let g:DoxygenToolkit_briefTag_pre="@desc function  "
let g:DoxygenToolkit_paramTag_pre="@params [type] "
let g:DoxygenToolkit_returnTag="@return "
"let g:DoxygenToolkit_blockHeader="--------------"
"let g:DoxygenToolkit_blockFooter="------------------"
"let g:DoxygenToolkit_authorName="Mathias Lorente"
"let g:DoxygenToolkit_licenseTag="My own license"
```

3.8.4 使用

注释操作都在 vim 命令行模式下操作：

（1） 注释类型（C/C++/// 或者，Python：##和#）：

在 vim 中，默认 C++注释为，但是如果你更喜欢使用///，只需要在你的配置文件。

vimrc 中添加如下语句：

let g:DoxygenToolkit_commentType="C++"。

（2） 许可：

在 vim 中，将光标放在将要写 doxygen 许可注释的那一行，然后，执行命令:DoxLic。这将会生成许可注释并将光标放置在刚才那一行之后。

（3） 作者：

在 vim 中，将光标放在想要添加 doxygen 作者注释的地方。然后执行命令:DoxAuthor。这将会生成一个框架，如果没有为其设置变量则将光标放置在@author 标签之后，或者放在在框架之后。

（4） 函数/类注释：

在 vim 中，将光标放置在函数头部那一行（或者函数的返回变量）或者类。然后执行命令:Dox。这将生成框架并且将光标放置在@brief 标签后。

（5）忽略代码片段（只有 C/C++）：

在 vim 中，如果你想要忽略所有在块中的代码片段，类似： #ifdef DEBUG ... #endif 你只需要执行以下命令:DoxUndoc **(DEBUG)！**

（5） 组：

在 vim 中，执行命令:DoxBlock 在后面的行中插入一个 doxygen 块。

3.9 vim 插件 tags 安装与配置

taglist 是一个用于显示定位程序中各种符号的插件，例如宏定义、变量

名、结构名、函数名这些东西 我们将其称之为符号(symbols)，而在 taglist 中将其称之为 tag。

显然，要想将程序文件中的 tag 显示出来，需要事先了解全部 tag 的信息，并将其保存在一个文件中，然后去解析对应的 tag 文件。

taglist 做的仅仅是将 tag 文件中的内容解析完后显示在 Vim 上而已。tag 扫描以及数据文件的生成则是由 ctags(Exuberant Ctags)这一工具完成的所以在使用 taglist 之前，你的电脑需要装有 ctags。

3.9.1 tags 的安装与配置

ctags 可以建立源码树的标签索引（标签就是一个标识符被定义的地方，如函数定义），使程序员在编程时能迅速定位函数、变量、宏定义等位置去查看原形。

Ctags 工具是用来遍历源代码文件生成 tags 文件，这些 tags 文件能被编辑器或其它工具用来快速查找定位源代码中的符号（tag/symbol），如变量名，函数名等。比如，tags 文件就是 Taglist 和 OmniCppComplete 工作的基础。

当前的 PHP 支持 ctags 发布(5.8)不支持当前的 PHP 版本的新功能名称空间、特征或旧功能接口。

有一个改进版的 https://ctags.io/

改进版 ctags 源码下载

https://github.com/shawncplus/phpcomplete.vim/raw/master/misc/ctags-5.8_better_php_parser.tar.gz

参考资料

https://github.com/shawncplus/phpcomplete.vim/wiki/Patched-ctags

http://blog.csdn.net/duguteng/article/details/7412652

ctags 相关网站

http://ctags.sourceforge.net/

以下是在 centos7 下 ctags 的下载安装和配置过程：

（1）ctags 改进版下载

软件：ctags-5.8_better_php_parser.tar

wget "https://github.com/shawncplus/phpcomplete.vim/raw/master/misc/ctags-5.8_better_php_parser.tar.gz" -O ctags-5.8_better_php_parser.tar.gz

（2）安装

```
tar xvf ctags-5.8_better_php_parser.tar.gz
./configure
make
make install
```

运行

ctags --version 将会看到 Exuberant Ctags Development 表示安装成功

（3）使用 ctags

cd /path/to/your/projects/root
运行 ctags -R --fields=+aimS --languages=php
参数说明
①-a：Access (or export) of class members; Adds the access field like: access: public

②-i：Inheritance information; Adds the inherits field like: inherits:RuntimeException,ExceptionInterface

③-m：Implementation information; Adds the implementation field like: implementation:abstract

-S
Signature of routine; Adds the signature field like: signature:($a, $b)

These extra fields will show up in the tag stack like this:
使用额外的参数字段将会看到类似下面的堆栈调试信息

6 F f handle
vendor/monolog/monolog/src/Monolog/Handler/NullHandler.php
class:Monolog\Handler::NullHandler access:public signature:(array $record)
public function handle(array $record)

3.10 目录管理工具 NERDtree

3.10.1 安装方式

用插件管理器 pathogen.vim 或者 Vundle 安装
cd ~/.vim/bundle
git clone https://github.com/scrooloose/nerdtree.git

3.10.2 启动 vim

自动打开 NERDtree
autocmd vimenter * NERDTree

3.10.3 无文件打开 vim 时自动开启 NERDtree

How can I open a NERDTree automatically when vim starts up if no files were specified?
autocmd StdinReadPre * let s:std_in=1
autocmd VimEnter * if argc() == 0 && !exists("s:std_in") | NERDTree | endif

3.10.4 映射快捷键

map <C-n> :NERDTreeToggle<CR>

3.10.5 如果只剩下 NERDTree 最后一个窗口，如何关闭 NERDTree

```
autocmd bufenter * if (winnr("$") == 1 && exists("b:NERDTree") && b:NERDTree.isTabTree()) | q | endif
```

3.10.6 显示不同颜色

（1）不同的扩展名，显示不同的颜色

Can I have different highlighting for different file extensions?

```
" NERDTress File highlighting
function! NERDTreeHighlightFile(extension, fg, bg, guifg, guibg)
 exec 'autocmd filetype nerdtree highlight ' . a:extension .' ctermbg='. a:bg .' ctermfg='. a:fg .' guibg='. a:guibg .' guifg='. a:guifg
 exec 'autocmd filetype nerdtree syn match ' . a:extension .' #^\s\+.*'. a:extension .'$#'
endfunction

call NERDTreeHighlightFile('jade', 'green', 'none', 'green', '#151515')
call NERDTreeHighlightFile('ini', 'yellow', 'none', 'yellow', '#151515')
call NERDTreeHighlightFile('md', 'blue', 'none', '#3366FF', '#151515')
call NERDTreeHighlightFile('yml', 'yellow', 'none', 'yellow', '#151515')
call NERDTreeHighlightFile('config', 'yellow', 'none', 'yellow', '#151515')
call NERDTreeHighlightFile('conf', 'yellow', 'none', 'yellow', '#151515')
call NERDTreeHighlightFile('json', 'yellow', 'none', 'yellow', '#151515')
call NERDTreeHighlightFile('html', 'yellow', 'none', 'yellow', '#151515')
call NERDTreeHighlightFile('styl', 'cyan', 'none', 'cyan', '#151515')
call NERDTreeHighlightFile('css', 'cyan', 'none', 'cyan', '#151515')
call NERDTreeHighlightFile('coffee', 'Red', 'none', 'red', '#151515')
call NERDTreeHighlightFile('js', 'Red', 'none', '#ffa500', '#151515')
call NERDTreeHighlightFile('php', 'Magenta', 'none', '#ff00ff', '#151515')
```

（2）改变默认的箭头

How can I change default arrows?

```
let g:NERDTreeDirArrowExpandable = '▸'
```

```
let g:NERDTreeDirArrowCollapsible = '▾'
```

4 centos 删除软件

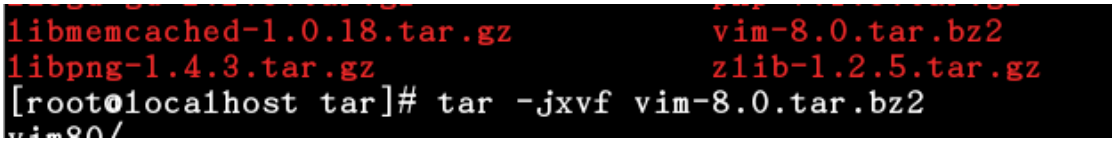
如果你带有 yum，可以直接 yum remove xxx
如果是 rpm 包，rpm -e xxx
tar 包的话需要你直接删除该文件或者 make uninstall xxx

whereis 查看软件安装路径

5 vim 常见问题及解决方法

5.1 vim 自带帮助不能使用

解压软件：



进入帮助文档目录
cd vim80/runtime/doc
复制帮助文档目录 doc 到软件安装录的 doc 中