

Centos7 软件包的管理与安装

软件包的管理

软件包的类型

rpm 二进制包-----》已经使用GCC 编译后的

tar 源码包-----》需要编译

RPM 概述：RPM 是 RPM Package Manager（RPM 软件包管理器）的缩写，这一文件格式名称虽然打上了 RedHat 的标志，但是其原始设计理念是开放式的，现在包括 OpenLinux、SUSE 以及 Turbo Linux 等 Linux 的分发版本都有采用，可以算是公认的行业标准了。

rpm 软件包的管理 rpm 包的获取方式

1)、Centos 系统镜像光盘

2)、网站 rpmfind.net

3)、比如安装 mysql、nginx 软件，我们可以去它的官方网站下载：<http://www.mysql.com> rpm 包格式的说明

```
1 [root@exercise1 ~]# ls /mnt/Packages/zsh-5.0.2-28.e17.x86_64.rpm
2 /mnt/Packages/zsh-5.0.2-28.e17.x86_64.rpm
3
4 zsh      -5.      0.      2-      28.
      e17.      x86_64.rpm
5
6 软件名  主版本号  次版本号  修订  release(第几次发布版本)  操
  作系统版本      软件包是64位包
7 #修订  指的是第几次修改bug。
8 #发布  指的是：第几次发布。发布时，可能只是对软件安装的默认参数做了修改而没有其它改动。
```

安装 rpm 软件

RPM 工具使用分为安装、查询、验证、更新、删除等操作

命令格式：rpm [参数] 软件包

参数:

- i 是install的意思, 安装软件包
- v 显示附加信息, 提供更多详细信息
- V 校验, 对已经安装的软件进行校验

-h --hash 安装时输出####标记

互动: rpm 使用时, 什么情况下使用软件包全名(有.rpm), 什么时候使用软件包名? (无.rpm)

全名: 在**安装和更新升级**时候使用

包名: 对**已经安装过**的软件包进行操作时, 比如查找已经安装的某个包, 卸载包等, 使用包名。它默认是去目录/var/lib/rpm下面进行搜索。当一个rpm包安装到系统上之后, **安装信息通常会保存在本地的/var/lib/rpm/目录下。**

从本地安装

1.需要进行挂载(mount)

```
1 [root@exercise1 ~]# mount /dev/cdrom /mnt/
2 mount: /dev/sr0 写保护, 将以只读方式挂载
3 [root@exercise1 ~]#
4
5 或者
6
7 [root@exercise1 ~]# mount /dev/sr0 /mnt/
8 mount: /dev/sr0 写保护, 将以只读方式挂载
9 [root@exercise1 ~]#
```

永久挂载与卸载临时挂载

```
1 方法一：卸载挂载源
2 [root@exercise1 ~]# umount /dev/cdrom
3
4 或
5
6 [root@exercise1 ~]# umount /dev/sr0
7
8 方法二：卸载挂载点
9 [root@exercise1 ~]# umount /mnt
```

永久：

```
[root@exercise1 ~]# vim /etc/fstab
```

#添加如下：

```
/dev/cdrom /mnt iso9660 defaults    0 0
```

```
[root@exercise1 ~]# df -h
文件系统      容量  已用  可用  已用% 挂载点
/dev/sda3      18G  1.2G   17G    7% /
devtmpfs       479M    0   479M    0% /dev
tmpfs          489M    0   489M    0% /dev/shm
tmpfs          489M  6.7M   482M    2% /run
tmpfs          489M    0   489M    0% /sys/fs/cgroup
/dev/sda1      197M   97M  100M   50% /boot
tmpfs          98M    0    98M    0% /run/user/0
/dev/sr0       4.3G  4.3G    0  100% /mnt
[root@exercise1 ~]# df -h
文件系统      容量  已用  可用  已用% 挂载点
/dev/sda3      18G  1.2G   17G    7% /
devtmpfs       479M    0   479M    0% /dev
tmpfs          489M    0   489M    0% /dev/shm
tmpfs          489M  6.7M   482M    2% /run
tmpfs          489M    0   489M    0% /sys/fs/cgroup
/dev/sda1      197M   97M  100M   50% /boot
tmpfs          98M    0    98M    0% /run/user/0
/dev/sr0       4.3G  4.3G    0  100% /mnt
```

2.开始安装

```
1 [root@exercise1 ~]# rpm -ivh /mnt/Packages/zsh-5.0.2-
  28.el7.x86_64.rpm
2 准备中...
  ##### [100%]
3 正在升级/安装...
4    1:zsh-5.0.2-28.el7
  ##### [100%]
5 [root@exercise1 ~]# cat /etc/shells
6 /bin/sh
7 /bin/bash
8 /sbin/nologin
9 /usr/bin/sh
10 /usr/bin/bash
11 /usr/sbin/nologin
12 /bin/zsh
13 [root@exercise1 ~]#
```

从网上下载直接安装 centos epel 扩展源

安装 centos epel 扩展 yum 源

```
1 [root@exercise1 ~]# rpm -ivh
  http://dl.fedoraproject.org/pub/epel/epel-release-latest-
  7.noarch.rpm
2 获取http://dl.fedoraproject.org/pub/epel/epel-release-latest-
  7.noarch.rpm
3 警告: /var/tmp/rpm-tmp.8hzUGV: 头V4 RSA/SHA256 Signature, 密钥 ID
  352c64e5: NOKEY
4 准备中...
  ##### [100%]
5 正在升级/安装...
6    1:epel-release-7-14
  ##### [100%]
7 [root@exercise1 ~]#
```

注: epel 源是对 centos7 系统中自带的 base 源的扩展。

rpm 查询功能

用法: rpm -q (query) 常与下面参数组合使用

- a (all) 查询所有已安装的软件包
- f (file) 系统文件名 (查询系统文件所属哪个软件包) , 反向查询
- i 显示已经安装的 rpm 软件包信息, 后面直接跟包名
- l (list) 查询软件包中文件安装的位置
- p 查询未安装软件包的相关信息, 后面要跟软件的命名
- R 查询软件包的依赖性

```
1 [root@exercise1 ~]# rpm -q zsh      --->查询指定的包是否安装
2 zsh-5.0.2-28.el7.x86_64
```

```
[root@exercise1 ~]# rpm -qa      --->查询所有已安装包
open-vm-tools-10.1.5-3.el7.x86_64
grub2-common-2.02-0.64.el7.centos.noarch
kexec-tools-2.0.14-17.el7.x86_64
.....
```

[root@exercise1 ~]# rpm -qa | grep vim --->查询所有已安装包中带 vim 关键字的包

```
vim-filesystem-7.4.629-8.el7_9.x86_64
vim-enhanced-7.4.629-8.el7_9.x86_64
vim-common-7.4.629-8.el7_9.x86_64
vim-minimal-7.4.160-2.el7.x86_64
[root@exercise1 ~]#
```

```
[root@exercise1 ~]# which find      #查看 find 命令的路径
/usr/bin/find
```

```
[root@exercise1 ~]# rpm -qf /usr/bin/find      #查询文件或命令属于哪个安装包
findutils-4.5.11-5.el7.x86_64
```

查询已经安装的 rpm 包的详细信息或作用 rpm -qi rpm包名

```
1 [root@exercise1 ~]# rpm -qi vim-common-7.4.629-8.el7_9.x86_64
2 Name           : vim-common
3 Epoch          : 2
4 Version        : 7.4.629
5 Release        : 8.el7_9
6 Architecture   : x86_64
```

```
7 Install Date: 2022年01月10日 星期一 08时49分49秒
8 Group       : Applications/Editors
9 Size        : 22155744
10 License     : Vim
11 Signature   : RSA/SHA256, 2020年12月18日 星期五 04时37分20秒, Key
    ID 24c6a8a7f4a80eb5
12 Source RPM  : vim-7.4.629-8.el7_9.src.rpm
13 Build Date   : 2020年12月16日 星期三 00时44分28秒
14 Build Host   : x86-01.bsys.centos.org
15 Relocations : (not relocatable)
16 Packager     : CentOS BuildSystem <http://bugs.centos.org>
17 Vendor       : CentOS
18 URL          : http://www.vim.org/
19 Summary      : The common files needed by any version of the
    VIM editor
20 Description  :
21 VIM (Visual editor iMproved) is an updated and improved
    version of the
22 vi editor. Vi was the first real screen-based editor for
    UNIX, and is
23 still very popular. VIM improves on vi by adding new
    features:
24 multiple windows, multi-level undo, block highlighting and
    more. The
25 vim-common package contains files which every VIM binary will
    need in
26 order to run.
27
28 If you are installing vim-enhanced or vim-X11, you'll also
    need
29 to install the vim-common package.
```

针对没有安装的 RPM 包，要加参数： -p

```
1 [root@home Packages]# rpm -qpi zip-3.0-11.el7.x86_64.rpm
2 警告: zip-3.0-11.el7.x86_64.rpm: 头V3 RSA/SHA256 Signature, 密钥
    ID f4a80eb5: NOKEY
3 Name          : zip
4 Version       : 3.0
5 Release       : 11.el7
6 Architecture: x86_64
7 Install Date: (not installed)
```

```
8 Group      : Applications/Archiving
9 Size       : 815173
10 License    : BSD
11 Signature   : RSA/SHA256, 2016年11月21日 星期一 05时04分58秒, Key
    ID 24c6a8a7f4a80eb5
12 Source RPM : zip-3.0-11.el7.src.rpm
13 Build Date  : 2016年11月06日 星期日 00时49分55秒
14 Build Host  : worker1.bsys.centos.org
15 Relocations : (not relocatable)
16 Packager    : CentOS BuildSystem <http://bugs.centos.org>
17 Vendor      : CentOS
18 URL         : http://www.info-zip.org/Zip.html
19 Summary     : A file compression and packaging utility
    compatible with PKZIP
20 Description :
21 The zip program is a compression and file packaging utility.
    Zip is
22 analogous to a combination of the UNIX tar and compress
    commands and
23 is compatible with PKZIP (a compression and file packaging
    utility for
24 MS-DOS systems).
25
26 Install the zip package if you need to compress files using
    the zip
27 program.
```

```
[root@exercise1 ~]# rpm -qpl /mnt/Packages/zip-3.0-11.el7.x86_64.rpm
#查看 rpm 安装后, 将生成哪些文件
/usr/bin/zip
/usr/bin/zipcloak
/usr/bin/zipnote
/usr/bin/zipsplit
/usr/share/doc/zip-3.0
/usr/share/doc/zip-3.0/CHANGES
/usr/share/doc/zip-3.0/LICENSE
/usr/share/doc/zip-3.0/README
/usr/share/doc/zip-3.0/README.CR
/usr/share/doc/zip-3.0/TODOL
/usr/share/doc/zip-3.0/WHATSNEW
```

```
/usr/share/doc/zip-3.0/WHERE
```

```
/usr/share/doc/zip-3.0/algorithm.txt
```

```
/usr/share/man/man1/zip.1.gz
```

```
/usr/share/man/man1/zipcloak.1.gz
```

```
/usr/share/man/man1/zipnote.1.gz
```

```
/usr/share/man/man1/zipsplit.1.gz
```

```
[root@exercise1 ~]# rpm -qR yum-3.4.3-154.el7.centos.noarch #查询软件
```

包的依赖性

```
/usr/bin/python
```

```
config(yum) = 3.4.3-154.el7.centos
```

```
cpio
```

```
diffutils
```

```
pygpgme
```

```
pyliblzma
```

```
python >= 2.4
```

```
python(abi) = 2.7
```

```
python-iniparse
```

```
python-sqlite
```

```
python-urlgrabber >= 3.10-8
```

```
pyxattr
```

```
rpm >= 0:4.11.3-22
```

```
rpm-python
```

```
rpmlib(CompressedFileNames) <= 3.0.4-1
```

```
rpmlib(FileDigests) <= 4.6.0-1
```

```
rpmlib(PayloadFilesHavePrefix) <= 4.0-1
```

```
yum-metadata-parser >= 1.1.0
```

```
yum-plugin-fastestmirror
```

```
rpmlib(PayloadIsXz) <= 5.2-1
```

```
[root@exercise1 ~]#
```

查看软件包内容是否被修改

rpm -V 包名

rpm -Vf 文件路径


```
1 [root@exercise1 ~]# which find
2 /usr/bin/find
3 [root@exercise1 ~]# rpm -qf /usr/bin/find
4 findutils-4.5.11-5.el7.x86_64
5 [root@exercise1 ~]# rpm -vf /usr/bin/find      #检查具体文件，没
   问题所以没有结果输出
6 [root@exercise1 ~]# echo aaa >> /usr/bin/find
7 [root@exercise1 ~]# rpm -vf /usr/bin/find
8 S.5....T.    /usr/bin/find
9 [root@exercise1 ~]# rpm -V findutils          #检查包
10 S.5....T.    /usr/bin/find
```

注：如果出现的全是点，表示测试通过 出现下面的字符代表某测试的失败：

5 — MD5 校验和是否改变，你也看成文件内容是否改变

S — 文件长度，大小是否改变

L — 符号链接，文件路径是否改变

T — 文件修改日期是否改变

D — 设备

U — 用户，文件的属主 G — 用户组

M — 模式 (包含讲可和文件类型)

? — 不可读文件

再后面的 c 文件名,它表示的是文件类型

c 配置文件

d 普通文件

g 不该出现的文件，意思就是这个文件不该被这个包所包含 l 授权文件 (license file) r 描述文件

实战：查看系统中所有的 rpm 包及安装的文件有没有被黑客修改

```
1 [root@exercise1 ~]# rpm -Va > /opt/rpm_check.txt
2 [root@exercise1 ~]# cat /opt/rpm_check.txt
3 .....T.   c /etc/bashrc
4 S.5....T.   c /root/.bashrc
5 S.5....T.   c /etc/sysconfig/authconfig
6 S.5....T.    /usr/bin/find
7 ....L..... c /etc/pam.d/fingerprint-auth
8 ....L..... c /etc/pam.d/password-auth
9 ....L..... c /etc/pam.d/postlogin
10 ....L..... c /etc/pam.d/smartcard-auth
11 ....L..... c /etc/pam.d/system-auth
12 遗漏      /var/run/wpa_supplicant
13 [root@exercise1 ~]#
```

> # 这个> 表示标准输出重定向。将 rpm -qa 输出到屏幕上的信息重定向到 rpm_check.txt 文件中。在文件中加一下这个参数描述

注：检验时参考了 [/var/lib/rpm](#) 目录下的 rpm 数据库信息

rpm 包卸载和升级

用法：rpm -e (erase) 包名

```
1 [root@exercise1 ~]# rpm -qa zsh
2 zsh-5.0.2-28.el7 .x86_64
3 [root@exercise1 ~]# rpm -e zsh
4 [root@exercise1 ~]#
5 [root@exercise1 ~]# rpm -qa zsh
6 参数： --nodeps 忽略依赖，建议在卸载时不要用 rpm 去卸载有依赖关系的包，
   应该用yum
7 [root@exercise1 ~]# rpm -e --nodeps lrzsz
8
9 升级：
10 [root@exercise1 ~]# rpm -Uvh /mnt/Packages/lrzsz-0.12.20-
   36.el7.x86_64.rpm      #因为升级时
11 会有一些依赖包要解决。 所以一般我们使用yum update 包 来升级。
12
13 例：解决 rpm 依赖关系：
14 [root@exercise1 ~]# rpm -ivh vim-common-7.4.160-
   2.el7.x86_64.rpm
15 警告：vim-common-7.4.160-2.el7.x86_64.rpm：头V3 RSA/SHA256
   Signature，密钥 ID f4a80eb5：NOKEY
16 错误：依赖检测失败：
17     vim-filesystem 被 vim-common-2:7.4.160-2.el7.x86_64 需要
```

```
18 [root@exercise1 ~]# rpm -ivh vim-filesystem-7.4.160-  
2.e17.x86_64.rpm  
19 [root@exercise1 ~]# rpm -ivh vim-common-7.4.160-  
2.e17.x86_64.rpm
```

YUM的使用

yum（全称为 Yellow dog Updater, Modified）是一个前端软件包管理器。基于 RPM 包管理，能够从指定的服务器自动下载 RPM 包并且安装，可以**自动处理依赖性关系，并且一次安装所有依赖的软件包**，无须繁琐地一次

次下载、安装。yum 提供了查找、安装、删除某一个、一组甚至全部软件包的命令，而且命令简洁而又好记

YUM：解决依赖关系问题，自动下载软件包，它是基于 C/S 架构

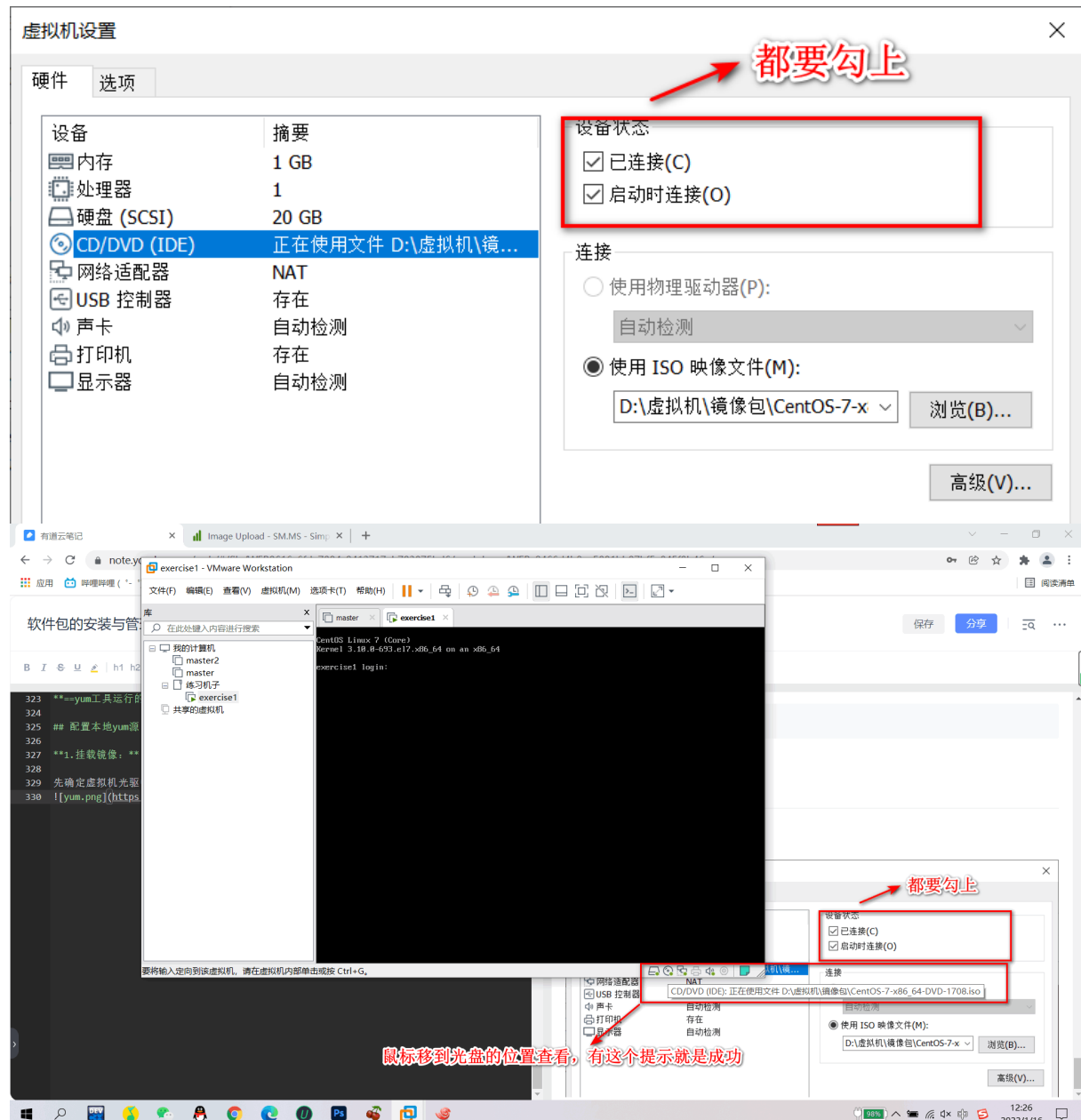
```
1 | C=client          S=ftp\http\file
```

yum工具运行的日志 `/var/log/yum.log`

配置本地yum源

1.挂载镜像：

先确定虚拟机光驱中有加载系统镜像



```
1 [root@exercise1 ~]# mount /dev/cdrom /mnt/
2 [root@exercise1 ~]# ls /mnt/
3 CentOS_BuildTag  EULA  images  LiveOS  repodata
   RPM-GPG-KEY-CentOS-Testing-7
4 EFI              GPL  isolinux  Packages  RPM-GPG-KEY-CentOS-
7  TRANS.TBL
5 [root@exercise1 ~]#
```

2.配置 yum 源文件:

```
1 [root@exercise1 ~]# mkdir /etc/yum.repos.d/bak
2 [root@exercise1 ~]# mv /etc/yum.repos.d/*
   /etc/yum.repos.d/bak  #把原来的.repo文件放在同一个目录下
3 [root@exercise1 ~]# ll /etc/yum.repos.d/bak/
4 总用量 36
```

```

5 -rw-r--r--. 1 root root 1664 8月 30 2017 CentOS-Base.repo
6 -rw-r--r--. 1 root root 1309 8月 30 2017 CentOS-CR.repo
7 -rw-r--r--. 1 root root 649 8月 30 2017 CentOS-
  Debuginfo.repo
8 -rw-r--r--. 1 root root 314 8月 30 2017 CentOS-
  fasttrack.repo
9 -rw-r--r--. 1 root root 630 8月 30 2017 CentOS-Media.repo
10 -rw-r--r--. 1 root root 1331 8月 30 2017 CentOS-Sources.repo
11 -rw-r--r--. 1 root root 3830 8月 30 2017 CentOS-Vault.repo
12 -rw-r--r--. 1 root root 1358 9月 5 01:37 epel.repo
13 -rw-r--r--. 1 root root 1457 9月 5 01:37 epel-testing.repo
14 [root@exercise1 ~]#

```

[root@exercise1 ~]# vim /etc/yum.repos.d/local.repo #必须以.repo结尾，插入以下内容

```

[local]
name=local
baseurl=file:///mnt
enable=1
gpgcheck=0
gpgkey=file:///mnt/RPM-GPG-KEY-CentOS-7

```

[root@exercise1 ~]# yum clean all #清空一下 yum 缓存

[root@exercise1 ~]# yum list #查看列表

[root@exercise1 ~]# yum repolist #显示仓库，统计各个仓库安装包数量

已加载插件：fastestmirror

Loading mirror speeds from cached hostfile

源标识	源名称
-----	-----

状态

loacl	local
-------	-------

3,894

repolist: 3,894

注:

[local] #yum 源名称，在本服务器上唯一的，用来区分不同的 yum 源

name= local #对 yum 源描述信息

baseurl=file:///mnt #yum 源的路径,提供方式包括 FTP(ftp://...)、HTTP(http://...)、本地(file:///...光盘挂载目录所在的位置)

enabled=1 #为 1，表示启用 yum 源；0 为禁用，若不设置，默认开启

gpgcheck=0

#为 1，使用公钥检验 rpm 包的正确性；0 为不校验

gpgkey=file:///mnt/RPM-GPG-KEY-CentOS-7 #指定进行 rpm 校验的公钥文件地址

安装vim

```
1 [root@home yum.repos.d]# yum install vim
```

下载命令

wget、curl

rz、sz

```
1 wget 下载资源，并且指定路径，同时指定文件名称 #如果下载时
   慢，可用迅雷下载完后，再上传liunx
2 -O: 另存为:
3
4 curl 是用来获取网页的源码信息的
5 -o: 另存为:
```

sz下载刚才的文件

```
1 语法:
2 sz 文件的绝对路径，或相对路径;
3 只可以是文件，不可以是文件夹;
4 问题:
5 1. 不支持超过4个G的文件;
6 2. 不支持断点续传;
7 3. 如果必须需要文件夹下载到本地，需要先压缩
```

rz是将windows的内容上传到liunx服务器;

1. 直接将文件拖进去;
2. 同样只可以是文件, 不可以是文件夹
3. 如果必须需要文件夹下载到本地, 需要先压缩

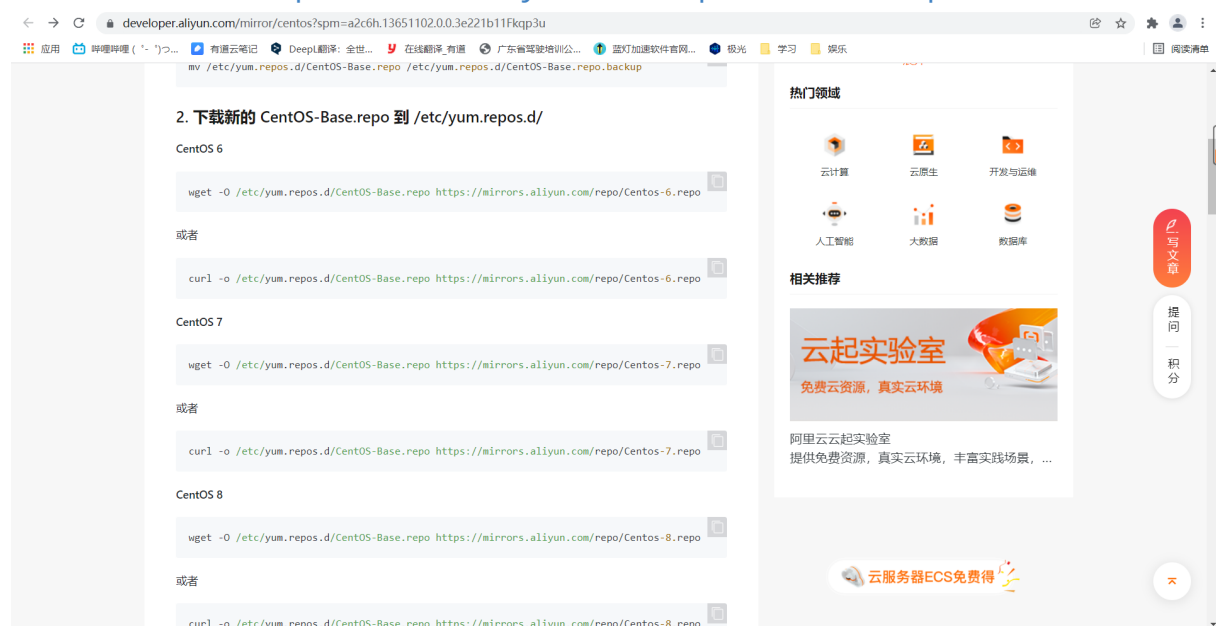
网络 yum 源

repo文件是Linux中yum源（软件仓库）的配置文件，通常一个repo文件定义了一个或者多个软件仓库的细节内容，例如我们将从哪里下载需要安装或者升级的软件包，repo文件中的设置内容将被yum读取和应用！

Centos 7 配置网络 yum 源

下载新的 CentOS-Base.repo 到 /etc/yum.repos.d/

可以选择**网易**的镜像源:<http://mirrors.163.com/.help/CentOS7-Base-163.repo> 或者是**阿里源**镜像 <https://mirrors.aliyun.com/repo/Centos-7.repo>



- 1 [root@exercise1 ~]# wget -O /etc/yum.repos.d/Centos-7.163.repo http://mirrors.163.com/.help/CentOS7-Base-163.repo
- 2
- 3 或者

```
[root@exercise1 ~]# wget -O /etc/yum.repos.d/CentOS-Base.repo https://mirrors.aliyun.com/repo/Centos-7.repo
--2022-01-16 13:51:47-- https://mirrors.aliyun.com/repo/Centos-7.repo
正在解析主机 mirrors.aliyun.com (mirrors.aliyun.com)... 119.147.41.242, 121.9.246.111, 113.113.101.248, ...
```

正在连接 mirrors.aliyun.com (mirrors.aliyun.com)|119.147.41.242|:443... 已连接。

已发出 HTTP 请求, 正在等待回应... 200 OK

长度: 2523 (2.5K) [application/octet-stream]

正在保存至: "/etc/yum.repos.d/CentOS-Base.repo"

100%[=====>] 2,523 --.-K/s 用时 0s

2022-01-16 13:51:47 (898 MB/s) - 已保存 "/etc/yum.repos.d/CentOS-Base.repo" [2523/2523])

```
[root@exercise1 ~]# ll /etc/yum.repos.d/
```

总用量 8

```
drwxr-xr-x. 2 root root 229 1月 16 13:31 bak
```

```
-rw-r--r--. 1 root root 2523 12月 26 2020 CentOS-Base.repo
```

```
-rw-r--r--. 1 root root 99 1月 16 13:40 local.repo
```

```
[root@exercise1 ~]#
```

- 1 **#wget** 下载文件, **-O** 将**wget** 下载的文件, 保存到指定的位置, 保存时可以重新起一个名字, 或者直接写一个要保存的路径, 这样还用原来的文件名。需要**yum**安装一下**wget**

#查看/etc/yum.repos.d/CentOS-Base.repo

```
[root@exercise1 ~]# vim /etc/yum.repos.d/CentOS-Base.repo
```

会看到其中一个仓库baseurl:

baseurl=[http://mirrors.aliyun.com/centos/\\$releasever/os/\\$basearch/](http://mirrors.aliyun.com/centos/$releasever/os/$basearch/)

[http://mirrors.aliyuncs.com/centos/\\$releasever/os/\\$basearch/](http://mirrors.aliyuncs.com/centos/$releasever/os/$basearch/)

[http://mirrors.cloud.aliyuncs.com/centos/\\$releasever/os/\\$basearch/](http://mirrors.cloud.aliyuncs.com/centos/$releasever/os/$basearch/)

#注: \$releasever 系统的版本的值等于操作系统版本

```
[root@exercise1 ~]# cat /etc/centos-release
```

CentOS Linux release 7.4.1708 (Core)

```
[root@exercise1 ~]#
```

\$basearch 等于: x86_64

排错: 如果下载 Centos-7.repo 后, 不能用

解决: 打开阿里云链接: <http://mirrors.aliyun.com/centos/> 找到 centos7 最新版本号, 如:


```
1 找到 7/ or 7.4.1708(本人使用系统版本) ==> x86_64 ==> Packages ==>
   把需要的rpm包下载放进之前/etc/yum.repos.d/bak/下
2
3  [root@localhost ~]# yum clean all    #清空一下 yum 缓存
4  [root@localhost ~]# yum list         #查看列表
```

yum 使用

请记住我们的世界里只有**upgrade!!!**

yum 常用操作:

```
1  [root@localhost ~]# yum list          #罗列各个仓库安装包
2  [root@localhost ~]# yum repolist      #统计各个仓库安装包数量
3  [root@localhost ~]# yum install -y httpd    #安装软件包, -y 安
   装过程需要询问的都默认为yes
4  [root@localhost ~]# yum -y update      #升级软件包, 改变软件设置和
   系统设置,系统版本内核都升级(高危操作, 慎用)
5  [root@localhost ~]# yum -y upgrade     #升级软件包, 不改变软件设置
   和系统设置, 系统版本升级, 内核不改变
6  [root@localhost ~]# yum provides /usr/bin/find    #查看命令是哪个
   软件包安装的
7  [root@localhost ~]# yum -y remove 包名    #卸载包
8  [root@localhost ~]# yum search keyword    #按关键字搜索软件包
9  [root@localhost ~]# yum info httpd        #查询 rpm 包作用
10 [root@localhost ~]# yum history            #查看历史
11 [root@localhost ~]# yum makecache          #生成yum缓存
12 [root@localhost ~]# yum makecache fast     #快速生成yum缓存
```

yum 报错, 注意的几个小问题:

1、确定光盘是否链接, 光盘是否挂载

2、配置文件中格式是否正确, 字母, 符号有没有少写,挂载点和配置文件中设置的是否一致

3、网络源需要联网, 操作和 RPM 类似, 只是会自动安装依赖项。

yum 安装开发工具软件包组

```

1 # yum grouplist      #查看有哪些软件包组
2 语法: yum groupinstall groupname
3 yum grouplist      #显示中文，如果想变成英文，则执行以下命令
4 [root@localhost ~]# echo $LANG
5 zh_CN.UTF-8
6 [root@localhost ~]# LANG=en_US.UTF-8
7 [root@localhost ~]# yum grouplist
8
9 测试:
10 [root@localhost ~]# yum remove gcc -y      #卸载开发工具软件组
    中的 gcc 包
11 [root@localhost ~]# yum groupinstall 'Development tools' -y
    #安装开发工具软件包组，安装这组软件包时，把 gcc 再安装上了

```

实战：创建本地yum仓库

下载rpm包到本地

使用阿里云的镜像仓库来同步到本地

1.使用reposync来进行同步阿里云镜像

reposync命令是一个python脚本。包含在yum-utils包中。

因此，我们如果要使用reposync命令的时候，需要安装yum-utils包。

使用以下命令：yum install -y yum-utils

reposync -r 仓库名（举例为base） -p 目标目录

```

1 #在主机exercise1操作
2 [root@exercise1 opt]# yum install -y yum-utils
3 [root@exercise1 ~]# yum repolist # 来查看各仓库名
4 已加载插件: fastestmirror
5 Loading mirror speeds from cached hostfile
6 * base: mirrors.aliyun.com
7 * extras: mirrors.aliyun.com
8 * updates: mirrors.aliyun.com
9 源标识

```

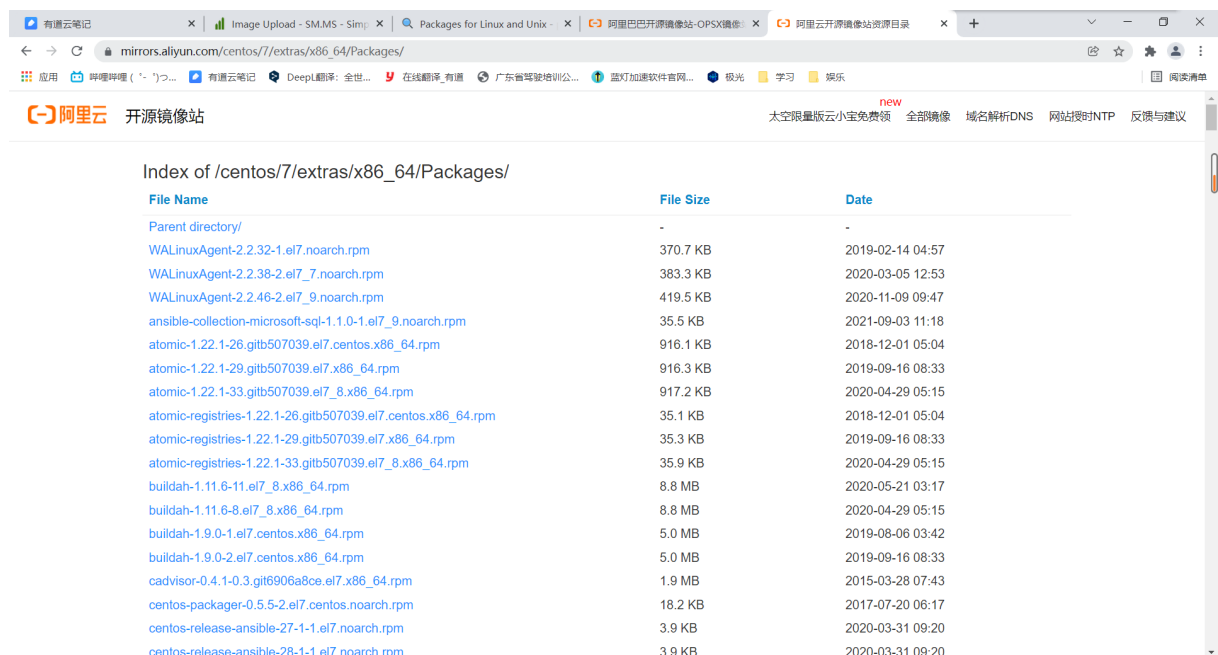
源名称

状态

```

10 base/7/x86_64 CentOS-7 -
    Base - mirrors.aliyun.com
    10,072
11 extras/7/x86_64 CentOS-7 -
    Extras - mirrors.aliyun.com
    500
12 local local
    3,894
13 updates/7/x86_64 CentOS-7 -
    Updates - mirrors.aliyun.com
    3,295
14 repolist: 17,761
15 [root@exercise1 ~]#
16
17 [root@exercise1 ~]# mkdir /opt/yum # 本地仓库目录
18 [root@exercise1 ~]# reposync -r extras -p /opt/yum/ #此时，就
    将镜像仓库extras(500个rpm)下载到了本地

```



File Name	File Size	Date
Parent directory/	-	-
WALinuxAgent-2.2.32-1.el7.noarch.rpm	370.7 KB	2019-02-14 04:57
WALinuxAgent-2.2.38-2.el7_7.noarch.rpm	383.3 KB	2020-03-05 12:53
WALinuxAgent-2.2.46-2.el7_9.noarch.rpm	419.5 KB	2020-11-09 09:47
ansible-collection-microsoft-sql-1.1.0-1.el7_9.noarch.rpm	35.5 KB	2021-09-03 11:18
atomic-1.22.1-26.gitb507039.el7.centos.x86_64.rpm	916.1 KB	2018-12-01 05:04
atomic-1.22.1-29.gitb507039.el7.x86_64.rpm	916.3 KB	2019-09-16 08:33
atomic-1.22.1-33.gitb507039.el7_8.x86_64.rpm	917.2 KB	2020-04-29 05:15
atomic-registries-1.22.1-26.gitb507039.el7.centos.x86_64.rpm	35.1 KB	2018-12-01 05:04
atomic-registries-1.22.1-29.gitb507039.el7.x86_64.rpm	35.3 KB	2019-09-16 08:33
atomic-registries-1.22.1-33.gitb507039.el7_8.x86_64.rpm	35.9 KB	2020-04-29 05:15
buildah-1.11.6-11.el7.x86_64.rpm	8.8 MB	2020-05-21 03:17
buildah-1.11.6-8.el7_8.x86_64.rpm	8.8 MB	2020-04-29 05:15
buildah-1.9.0-1.el7.centos.x86_64.rpm	5.0 MB	2019-08-06 03:42
buildah-1.9.0-2.el7.centos.x86_64.rpm	5.0 MB	2019-09-16 08:33
cadvisor-0.4.1-0.3.git6906a8ce.el7.x86_64.rpm	1.9 MB	2015-03-28 07:43
centos-packager-0.5.5-2.el7.centos.noarch.rpm	18.2 KB	2017-07-20 06:17
centos-release-ansible-27-1-1.el7.noarch.rpm	3.9 KB	2020-03-31 09:20
centos-release-ansible-28-1-1.el7.noarch.rpm	3.9 KB	2020-03-31 09:20

2.可以只把需要的rpm包下载到本地

使用yum命令加上参数

--downloadonly: 只下载

--downloadaddir: 指定下载目录

```

1 #在主机exercise1操作
2 [root@teach opt]#yum install zlib-devel.i686 --downloadonly -
    -downloadaddir=/opt/yum/extras/Packages/

```

生成yum仓库数据文件信息(repodata信息)

1.createrepo生成repodata信息

```
1 #在主机exercise1操作
2 [root@exercise1 yum]# yum install -y createrepo #需要yum安装 creatrepo
3 [root@exercise1 yum]# createrepo /opt/yum/ #使用完成后会在/opt/yum/目录里面生成repodata, 这个文件里面存放的就是仓库的各项信息
```

2.检查本地仓库信息是否正常

1): 配置本地yum仓库

```
1 #在主机exercise1操作
2 [root@exercise1 yum]# vim /etc/yum.repos.d/local.repo #在原来的基础上添加多一个local2仓库
3 [local2]
4 name=local2
5 baseurl=file:///opt/yum
6 enabled=1
7 gpgcheck=0
8
9 [root@exercise1 ~]# yum clean all #清缓存
10 [root@exercise1 ~]# yum list #罗列各个仓库安装包
11 [root@exercise1 ~]# yum repolist #统计各个仓库安装包数量
```

2): 检查本地仓库信息

```
1 [root@exercise1 ~]# yum repoinfo local # 输出正常表示可以进行正常使用
```

配置ftp服务, 提供局域网下载

```

1 #在主机exercise1操作
2 [root@exercise1 ~]# yum install -y vsftpd
3 [root@exercise1 ~]# systemctl stop firewalld #关闭防火墙
4 [root@exercise1 ~]# systemctl disable firewalld #关闭开机自启
5 [root@exercise1 ~]# setenforce 0 #临时关闭selinux安全策略
6 [root@home ~]# vim /etc/selinux/config #永久关闭selinux安全策略
7 修改: SELINUX=enforcing
8 为: SELINUX=disabled
9 [root@exercise1 ~]# systemctl start vsftpd
10 [root@exercise1 ~]# mv /opt/yum/ /var/ftp/

```

默认主目录/var/ftp=ftp://主机IP

测试局域网其他机器是否能够访问

在另一台机上执行exercise2

```

1 [root@exercise2 ~]# vim /etc/yum.repos.d/local.repo
2 #在原来的基础上添加多一个local3仓库
3 [local3]
4 name=local3
5 baseurl=ftp://192.168.119.142/yum
6 enabled=1
7 gpgcheck=0
8
9 [root@exercise2 ~]# yum repolist
10 已加载插件: fastestmirror
11 Loading mirror speeds from cached hostfile
12 * base: mirrors.aliyun.com
13 * extras: mirrors.aliyun.com
14 * updates: mirrors.aliyun.com
15 源标识 源名称 状态
16 base/7/x86_64 CentOS-7 -
Base - mirrors.aliyun.com
10,072
17 extras/7/x86_64 CentOS-7 -
Extras - mirrors.aliyun.com
500
18 local local
3,894

```

```

19 local3                                     local3
500
20 updates/7/x86_64                          CentOS-7 -
updates - mirrors.aliyun.com
3,295
21 repolist: 18,261
22 [root@exercise2 ~]#
23 #可以连通，证明可以正常使用

```

ftp://ip地址/yum ==> /var/ftp/yum

本地yum仓库更新

当本地仓库有其他包加入或者同步网络yum源的时候有变动，就需要更新本地yum仓库

在exercise1执行：

`reposync -r 仓库名 -p /opt/yum # 同步并更新`

`createrepo --update /opt/yum # 更新repopdata信息`

`yum clean all && yum repolist # 更新缓存`

在exercise2执行：

`yum clean all && yum repolist # 更新缓存`

源码安装 nginx

1. 编译环境如 gcc 和 gcc-c++编译器，make

```

1 yum -y install gcc gcc-c++ make zlib-devel pcre pcre-devel
  openssl-devel      #pcre: 支持正则表达式，地址重写 rewrite

```

2. 准备软件 ： nginx1.18 去官网(<http://nginx.org/en/download.html>)复制对应版本的下载链接

```
[root@localhost ~]# cd /opt
[root@exercise1 opt]# wget https://nginx.org/download/nginx-1.18.0.tar.gz
[root@exercise1 opt]# tar -zxvf nginx-1.18.0.tar.gz
[root@exercise1 opt]# cd nginx-1.18.0
```

3. 部署 Nginx

安装路径为--prefix /usr/local/nginx

用户组为 --user --group nginx

指定配置文件 --conf-path /etc/nginx/nginx.conf

指定错误日志路径--error-log-path /var/log/nginx/error.log

指定访问日志路径--http-log-path /var/log/nginx/access.log

--with-threads enable thread pool support

--with-http_ssl_module 启用ngx_http_ssl_module支持（使支持https请求）

--with-http_realip_module 启用ngx_http_realip_module支持（这个模块允许从请求标头更改客户端的IP地址值，默认为关）

--with-http_image_filter_module 启用ngx_http_image_filter_module支持（传输JPEG/GIF/PNG 图片的一个过滤器）

--with-http_sub_module 启用ngx_http_sub_module支持（允许用一些其他文本替换nginx响应中的一些文本）

--with-http_flv_module 启用ngx_http_flv_module支持（提供寻求内存使用基于时间的偏移量文件）

--with-http_gzip_static_module 启用ngx_http_gzip_static_module支持（在线实时压缩输出数据流）

--with-http_gunzip_module enable ngx_http_gunzip_module

--with-http_stub_status_module 启用ngx_http_stub_status_module支持（获取nginx自上次启动以来的工作状态）

--with-http_v2_module enable ngx_http_v2_module

#根据官网对应文档(<http://nginx.org/en/docs/configure.html>)来查看对应参数进行预编译

```
[root@exercise1 nginx-1.18.0]# ./configure --prefix=/usr/local/nginx --
user=nginx --group=nginx --conf-path=/etc/nginx/nginx.conf --error-log-
path=/var/log/nginx/error.log --http-log-path=/var/log/nginx/access.log --
with-threads --with-http_ssl_module --with-http_realip_module --with-
http_image_filter_module --with-http_sub_module --with-http_flv_module --
```

```
with-http_gzip_static_module --with-http_gzip_static_module --with-  
http_gunzip_module --with-http_stub_status_module --with-  
http_v2_module
```

其中需要解决依赖，当不知道依赖包完整名称是什么的时候可以来这里找一下 <http://pkgs.org/> ==> 宝藏网站，利用它查找包

```
1 [root@exercise1 nginx-1.18.0]# make -j 4    #使用-j 4 指定 4 核心  
   CPU 编译，提升速度  
2 [root@exercise1 nginx-1.18.0]# make install
```

详解源码安装 3 把斧

```
1 # ./configure  
2 ./configure --help          #可以提供预编译支持帮助  
3 a. 指定安装路径，例如 --prefix=/usr/local/nginx  
4 b. 启用或禁用某项功能，例如 --enable-ssl, --disable-filter --  
   with-http_ssl_module  
5 c. 和其它软件关联，例如--with-pcre  
6 d. 检查安装环境，例如是否有编译器 gcc ， 是否满足软件的依赖需求  
7 最终生成: Makefile  
8 #make -j 4                  #按 Makefile 文件编译，可以使用-j 4 指定 4 核心  
   CPU 编译，提升速度  
9 #make install              #按 Makefile 定义的文件路径安装  
10 #make uninstall           #按 Makefile 定义的文件路径删除  
11 # make clean //清除上次的 make 命令所产生的 object 和 Makefile 文  
   件。使用场景：当需要重新执行 configure 时，需要执行 make clean
```

删除源码包

安装完，删除： **make uninstall**

有时删除不干净，所以建议大家安装时，在 configure 步骤添加一个：--prefix 参数。这样删除或备份时， 直接对删除--prefix 指定的安装目录操作就可以了。

实战 2：源码编译出错的 5 种完美解决方法

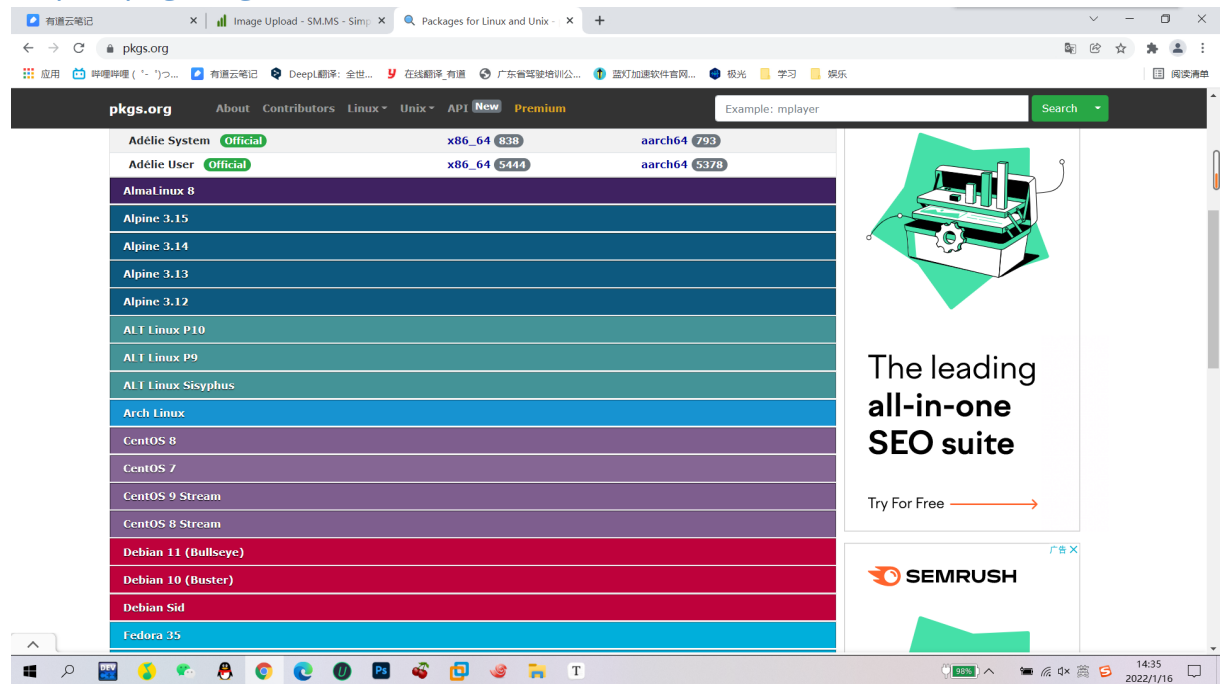
常见解决方法：

共5种方法

```
1 方法 1: [root@exercise1 ~]# rpm -ivh gcc^C    #按两下 tab 键。 一  
   般情况，ext2fs 就是 要安装的软件包的名字开头。如果存在 自动补全  
2 方法 2: [root@exercise1 ~]# ls *gcc*         #查找完整关键字  
3 方法 3: [root@exercise1 ~]# ls *cc*          #查找部分关键字
```


方法 4: 终极大招

<https://pkgs.org/>



- 1 方法 5: 使用 yum 去搜索
- 2 [root@exercise1 ~]# yum search gcc

总结,软件安装方法特点:

rpm + yum : 方便, 软件版本低。稳定性好、管理方便。性能稍差。

源码编译安装: 麻烦, 软件版本新, 可以定制。稳定性稍差、管理稍差。性能好。

源码编译安装: 主要是安装 LAMP 或 LNMP 架构时, 我们会用