index

APT软件源仓库管理

Author:sandylaw

Email :freelxs@gmail.com

Date :2020-12-24

tags : linux, apt

目录

Debian软件源结构

对于典型的 HTTP 访问，软件源在 `/etc/apt/sources.list` 文件中指定。

```
deb http://deb.debian.org/debian签名/ buster main contrib non-free
deb-src http://deb.debian.org/debian/ buster main contrib non-free
```

对于UOS的软件源，配置类似，例如专业版的软件源：

```
deb https://professional-packages.chinauos.com/desktop-professional eagle main
contrib non-free #此域名为官方主仓库，需要通过授权管理工具激活，方可使用
```

> 详见内网仓库说明

`/etc/apt/sources.list` 的含义在 `sources.list(5)` 中进行了描述，下面是一些要点[1]。

- deb 的那行定义了二进制软件包。
- deb-src 的那行定义了源代码软件包。
- 第一个参数是 Debian 档案库的根 URL，常见支持http、https协议。ftp需要单独配置。
- 第二个参数是发行版名称：可以使用套件名或代号(codename)。
- 第三个和之后的参数是 Debian 档案库的有效档案库范围名称。
  - main 遵从 Debian 自由软件指导方针（DFSG），并且不依赖于 non-free
  - contrib 遵从 Debian 自由软件指导方针（DFSG），但依赖于 non-free
  - non-free 不遵从 Debian 自由软件指导方针（DFSG）

> 更新仓库：

```
sudo apt update #取回更新的软件包列表信息
sudo apt upgrade #更新系统
sudo apt dist-upgrade #发行版升级
```

Deb包签名

什么是GPG[2]

要了解什么是GPG，就要先了解PGP。

1991年，程序员Phil Zimmermann为了避开政府监视，开发了加密软件PGP。这个软件非常好用，迅速流传开来，成了许多程序员的必备工具。但是，它是商业软件，不能自由使用。所以，自由软件基金会决定，开发一个PGP的替代品，取名为GnuPG。这就是GPG的由来。

GPG有许多用途，本文主要介绍文件加密。至于邮件的加密，不同的邮件客户端有不同的设置，请参考Ubuntu网站的介绍。

本文的使用环境为Linux命令行。如果掌握了命令行，Windows 或 Mac OS 客户端，就非常容易掌握。GPG并不难学，学会了它，从此就能轻松传递加密信息。建议读者一步步跟着教程做，对每条命令都自行测试。

GPG有两种安装方式。

- 可以下载源码，自己编译安装。

```
./configure
make
make install
```

- 也可以安装编译好的二进制包。

```
sudo apt-get install gnupg
```

- 安装完成后，键入下面的命令：

```
gpg --help
```

如果屏幕显示GPG的帮助，就表示安装成功。

生成随即密码

随机 16 位密码：`openssl rand -base64 16`

生成GPG KEY

运行命令：`gpg --full-gen-key`

按照提示输入姓名、邮箱，确认，有效期，输入密码，会在`~/.gnupg/openpgp-revocs.d/`目录下生成`.rev`的 key 文件。

查看GPG密钥

list-keys参数列出系统中已有的密钥.

> gpg --list-keys

显示结果如下:

> /home/uos/.gnupg/pubring.kbx

> pub rsa4096 2020-07-30 [SCEA]
> 9245BF9CB425D2D241C23542C28CB811A1B7D01C
> uid [未知] devicepackages (devicepackages) devicepackages@uniontech.com
> sub rsa4096 2020-07-30 [SEA]

第一行显示公钥文件名（pubring.gpg），第二行显示公钥特征（4096位，Hash字符串和生成时间），第三行显示"用户ID"，第四行显示私钥特征。

> `gpg -K`列出私钥，`gpg -k`列出公钥。

如果你要从密钥列表中删除某个密钥，可以使用delete-key参数。

> gpg --delete-key [用户ID]
>
> `--delete-secret-keys`删除私钥，`--delete-secret-and-public-keys`删除公钥和私钥。

导出GPG公钥

公钥文件（.gnupg/pubring.gpg）以二进制形式储存，armor参数可以将其转换为ASCII码显示。

> gpg --armor --output public-key.txt --armor --export [用户ID]

"用户ID"指定哪个用户的公钥，output参数指定输出文件名（public-key.txt）。

对 Deb 包进行签名[3]

首先安装deb包签名工具：`apt-get install dpkg-sig`

使用`dpkg-buildpackage -us -uc`生成无前面的软件包后，进行手动签名：

`dpkg-sig -k 9245BF9C --sign builder mypackage_0.1.2_amd64.deb`

验证签名：

`dpkg-sig --verify mypackage_0.1.2_amd64.deb`

Apach介绍

Apache简介

Apache HTTP Server（简称Apache）是Apache软件基金会的一个开放源码的网页服务器，可以在大多数计算机操作系统中运行，由于其多平台和安全性被广泛使用，是最流行的Web服务器端软件之一[4]。

Apache2安装与配置

`sudo apt install apache2`

默认网站根目录在`/var/www/`文件夹下，可以通过创建软链接的方式把实际网站的目录链接到此目录下。

配置文件在`/etc/apache2/`文件夹下，主要有：

- apache2.conf 主配置文件，搭建仓库不需修改。
- conf-available 文件夹下是各个网站物理文件配置
- confi-enabled 文件夹下启用的配置，是上面的链接
- site-available 文件夹下是各个网站的配置
- site-enabled 文件夹下是启用的网站配置，是上面的链接

修改配置后，记得重启服务。

```
sudo a2enconf repos
sudo a2ensite repos
sudo apache2ctl configtest
sudo systemctl daemon-reload
sudo systemctl restart apache2.service
```

Vsftp介绍

Vsftp简介

vsftpd (“Very Secure FTP Daemon“) 是一个为 UNIX 类系统开发的轻量，稳定和安全的 FTP 服务器端。

Vsftp安装与配置

`sudo apt install vsftpd`

vsftpd 的大多数配置都可以通过编辑 /etc/vsftpd.conf 文件实现。 该文件本身自带大量注释说明，所以这一章节只就一些重要的配置予以说明。有关所有可用选项和文档，请参阅 vsftpd.conf(5) 手册页。默认情况下，由 /srv/ftp 提供文件[5]。

允许匿名下载相关配置项：

```
anonymous_enable=YES
no_anon_password=YES
anon_root=/srv/ftp/
```

Reprepro介绍

reprepro[6] 是用于管理 deb 格式软件包，生成用于分发的仓库管理工具。 支持 .dsc/.deb/.udeb 等格式；会根据配置生成 Packages/Sources 文件以及压缩版本， 并对 Release (根据配置还生成 Release.gpg) 。

安装

`sudo apt install reprepro`

配置

- 配置GPG密钥
- 使用Apache搭建http服务器
- 配置Reprepro(conf/)

详细文档参考man 5手册(请先通读一遍)。

专用设备软件仓库的搭建

专用设备软件源仓库规划

- 专用设备版软件源仓库名称

依据产品线定义仓库名称：device

- 专用设备版软件源仓库分支管理

同一个产品线不同的维护分支采用 codename 加上维护版本定义。

GUI 产品：mars mars/sp1 mars/sp2

CLI 产品：venus venus/sp1 venus/sp2

- 专用设备版软件源仓库分类
- 内网unstable仓库
  `deb http://10.8.0.113/unstable/device/ CODENAME main contrib non-free`
- 内网stable主仓库
  `deb deb http://10.8.0.113/stable/device/ CODENAME main contrib non-free`
- 外网发布仓库
  `deb https://device-packages.chinauos.com/device/ CODENAME main contrib non-free`

安装依赖软件包

`sudo apt install openssl gpg dpkg-sig reprepro apache2 cron moreutils httrack rrdtool logrotate apt-mirror -y`

生成GPG密钥

通过keydetails文件的形式生成GPG密钥并导出ASCII公钥，注意相关变量：

```
cat > keydetails << EOF
    %echo Generating a basic OpenPGP key
    Key-Type: RSA
    Key-Length: 4096
    Subkey-Type: RSA
    Subkey-Length: 4096
    Name-Real: $GPGNAME
    Name-Comment: $GPGNAME
    Name-Email: $GPGEMAIL
    Expire-Date: 0
    Passphrase: $PASSPHRASE
    %no-ask-passphrase
    %no-protection
    %pubring pubring.kbx
    %secring trustdb.gpg
```

```
    # Do a commit here, so that we can later print "done" :-)
    %commit
    %echo done
EOF
gpg --verbose --batch --gen-key keydetails
#echo "Generate the ASCII Format Public Key"
gpg --output "${GPGEMAIL}".gpg.key --armor --export "$GPGEMAIL"
```

准备仓库所需磁盘

假设已准备好1T以上的磁盘，格式化为`ext4`分区，并挂载到`/data`。

```
sudo ln -s /data/repos /var/www/repos
```

为Reprepro配置Apache

- `/data/`下创建文件夹`repos`,并链接到`/var/www/repos`
- 修改`/etc/apache2/sites-available/000-default.conf`中的默认80端口的网站根目录为`/var/www/html`,端口为其他端口，比如99。

```
Listen 99
<VirtualHost *:99>
    DocumentRoot /var/www/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

- 为APT仓库新建虚拟主机配置`/etc/apache2/sites-available/repos.conf`:

```
<VirtualHost *:80>
        DocumentRoot /var/www/repos
        ErrorLog /error.log
        CustomLog /access.log combined
</VirtualHost>
```

- 修改`/etc/apache2/apache2.conf`中的`ServerName`配置为:

```
ServerName localhost
```

- 为Reprepro新建配置`/etc/apache2/conf-available/repos.conf`:

> 其中：仅对外部暴露dists\pool，其他目录予以隐藏处理。

```
<Directory /var/www/repos/ >
     # We want the user to be able to browse the directory manually
     Options Indexes FollowSymLinks Multiviews
     Order allow,deny
     Allow from all
 </Directory>
 # This syntax supports several repositories, e.g. one for Debian, one for
Ubuntu.
 # Replace * with debian, if you intend to support one distribution only.
 <Directory "/var/www/repos/*/*/db/">
     Order allow,deny
```

```
    Deny from all
</Directory>
<Directory "/var/www/repos/*/*/conf/">
    Order allow,deny
    Deny from all
</Directory>
<Directory "/var/www/repos/*/*/listmorguedirs/">
    Order allow,deny
    Deny from all
</Directory>
<Directory "/var/www/repos/*/*/logs/">
    Order allow,deny
    Deny from all
</Directory>
 <Directory "/var/www/repos/*/*/morguedir/">
    Order allow,deny
    Deny from all
</Directory>
 <Directory "/var/www/repos/script/">
    Order allow,deny
    Deny from all
</Directory>
<Directory "/var/www/repos/apt-mirror/">
   Order allow,deny
   Deny from all
</Directory>
<Directory "/var/www/repos/*/*/incoming/">
    Order allow,deny
    Deny from all
</Directory>
```

- 重启Apache服务

```
sudo a2enconf repos
sudo a2ensite repos
sudo apache2ctl configtest
sudo systemctl daemon-reload
sudo systemctl restart apache2.service
```

配置Reprepro

- APT软件源目录层次

```
├── stable
│   └── device
│       ├── conf
│       ├── db
│       ├── dists
│       ├── lists
│       ├── logs
│       ├── morguedir
│       └── pool
└── unstable
    ├── device
    │   ├── conf
    │   ├── db
```

```
    │   ├── dists
    │   ├── logs
    │   ├── morguedir
    │   └── pool
    └── devicepackages.key
```

- 分支：stable、unstable
- 产品：device
- conf: 配置目录，主要包括distributions、updates
- logs:日志
- morguedir:deb删除后的备份
- dists：软件源元数据，下面分codename

```
dists/
├── mars            #GUI代号
│   ├── 1010        #mars/1010
│   ├── contrib     #属于自由软件但多半依赖非自由 （ non-free ） 软件
│   ├── InRelease   #内联签名的Release
│   ├── main        #最基本及主要且符合自由软件规范的软件 （ packages ）
│   ├── non-free    #不属于自由软件范畴的软件
│   ├── Release     #档案库描述和完整性信息
│   └── Release.gpg #"Release" 文件的签名文件，使用档案库密钥签名
└── venus           #CLI代号
    ├── contrib
    ├── InRelease
    ├── main
    ├── non-free
    ├── Release
    └── Release.gpg
```

- pool: 软件包的物理地址。

```
pool/
├── contrib
├── main
└── non-free
```

软件包均放进一个巨大的 "池子(pool)"， 按照源码包名称分类存放。为了方便管理, pool 目录下按属性再分类("main", "contrib" 和 "non-free")，分类下面再按源码包名称的首字母归档. 这些目录包含的文件有：运行于各种系统架构的二进制软件包， 生成这些二进制软件包的源码包。

- distributions配置

```
Origin: UOS Device
Label: Device
Suite: stable
Codename: mars
Version: 2020
Update: 1000
Architectures: i386 amd64 arm64 mips64el sw_64 source
Components: main contrib non-free
UDebComponents: main
Contents: percomponent nocompatsymlink .bz2
SignWith: devicepackages@uniontech.com
Description: UOS Device Packages
```

```
DebIndices: Packages Release . .gz /usr/bin/rredtool
Log: uos_mars.log

Origin: UOS Device
Label: Device
Suite: stable
Codename: mars/1010
Version: 2020
Update: mars/1010
Architectures: i386 amd64 arm64 mips64el sw_64 source
Components: main contrib non-free
UDebComponents: main
Contents: percomponent nocompatsymlink .bz2
SignWith: devicepackages@uniontech.com
Description: UOS Device Packages
DebIndices: Packages Release . .gz /usr/bin/rredtool
Log: uos_mars-1010.log
```

其中:

- DebIndices: 是借用了rredtool程序（来自于rrdtool软件包）生成更新deb的diff日志文件，比如位于`stable/device/dists/venus/main/binary-amd64/Packages.diff/`目录下。
- Update: 需要配合下面将要介绍的updates配置文件来完成从上游仓库指定配置来更新此仓库对应的codename。

> i386是deepin-wine需要

- updates配置

```
Name: 1010
#Suite: stable
Suite: eagle/sp3
Architectures: i386 amd64 arm64 mips64el sw_64 source
Components: main contrib non-free
Method: file:///data/apt-mirror-desktop/mirror/pools.uniontech.com/desktop-professional/
VerifyRelease: blindtrust

Name: mars
Suite: mars
Architectures: i386 amd64 arm64 mips64el sw_64 source
Components: main contrib non-free
Method: http://127.0.0.1/unstable/device/
VerifyRelease: blindtrust
```

其中:

- Method: 是上游仓库地址
- Suite: 是上游仓库的codename
- Name: 是本仓库要更新的配置名称，与distributions配置文件中的update:字段名称一致。

管理APT仓库

- 添加软件到仓库

```
find "$DEBDIR" -name "*.deb" -exec sudo GNUPGHOME=/home/"$TUSER"/.gnupg
reprepro -C "$COMP" --ask-passphrase -Vb "$REPOSDIR" includedeb "$CODENAME"
{} +
```

    - COMP: 选择一个分类 main contrib non-free
    - REPODIR: 比如 `/data/repos/unstable/device/`
    - CODENAME:预要添加到仓库的codename
    - GNUPAHOME： 指定GPG目录，对deb包签名后再添加到仓库
  - includedeb：添加deb包

```
sudo  GNUPGHOME=/home/"$TUSER"/.gnupg reprepro -C "$COMP" --ask-passphrase
-Vb "$REPOSDIR" includedsc "$CODENAME" "$dsc"
```

    - dsc: 通过添加dsc文件来添加源码包到仓库 #“DSC”是Debian Source Control的首字母缩写。
    - includedsc:添加源码包
- 列出仓库的软件

```
sudo GNUPGHOME=/home/uos/.gnupg reprepro --ask-passphrase -Vb "$REPOSDIR"
list "$CODENAME"
```

- 从仓库删除软件

```
sudo GNUPGHOME=/home/uos/.gnupg reprepro --morguedir
+b/morguedir/"$CODENAME" --ask-passphrase -Vb "$REPOSDIR" remove
"${CODENAME}" packagename
```

> 删除软件并删除源码包

- 复制软件

```
sudo GNUPGHOME=/home/"$TUSER"/.gnupg reprepro -C "$COMP" --ask-passphrase -
Vb "$REPOSDIR" copy "$_dest" "$CODENAME" packagename
```

    - _dest: 目标codename
    - codename：源codename
    - copy：复制deb包
    - copysrc：复制源码包

同步上游仓库

使用`apt-mirror`来镜像上游仓库。

- 在`/data/`目录下创建上游仓库所需的镜像目录，比如`apt-mirror-desktop`
- 配置`/etc/apt/mirror.list`:

```
############# config ##################
#
# set base_path    /var/spool/apt-mirror
```

```
#
# set mirror_path  $base_path/mirror
# set skel_path    $base_path/skel
# set var_path     $base_path/var
# set cleanscript $var_path/clean.sh
# set defaultarch  <running host architecture>
# set postmirror_script $var_path/postmirror.sh
# set run_postmirror 0
set base_path /data/apt-mirror-desktop
set nthreads     20
set _tilde 0
#
############# end config ##############

# mirror additional architectures
deb-amd64 http://pools.uniontech.com/desktop-professional eagle/sp2 main
contrib non-free
deb-amd64 http://pools.uniontech.com/desktop-professional eagle/sp2
main/debian-installer
deb-arm64 http://pools.uniontech.com/desktop-professional eagle/sp2 main
contrib non-free
deb-arm64 http://pools.uniontech.com/desktop-professional eagle/sp2
main/debian-installer
deb-i386 http://pools.uniontech.com/desktop-professional eagle/sp2 main contrib
non-free
deb-i386 http://pools.uniontech.com/desktop-professional eagle/sp2 main/debian-
installer
deb-mips64el http://pools.uniontech.com/desktop-professional eagle/sp2 main
contrib non-free
deb-mips64el http://pools.uniontech.com/desktop-professional eagle/sp2
main/debian-installer
deb-sw_64 http://pools.uniontech.com/desktop-professional eagle/sp2 main
contrib non-free
deb-sw_64 http://pools.uniontech.com/desktop-professional eagle/sp2
main/debian-installer
deb-src http://pools.uniontech.com/desktop-professional eagle/sp2 main contrib
non-free

deb-amd64 http://pools.uniontech.com/desktop-professional eagle/sp3 main
contrib non-free
deb-amd64 http://pools.uniontech.com/desktop-professional eagle/sp3
main/debian-installer
deb-arm64 http://pools.uniontech.com/desktop-professional eagle/sp3 main
contrib non-free
deb-arm64 http://pools.uniontech.com/desktop-professional eagle/sp3
main/debian-installer
deb-i386 http://pools.uniontech.com/desktop-professional eagle/sp3 main contrib
non-free
deb-i386 http://pools.uniontech.com/desktop-professional eagle/sp3 main/debian-
installer
deb-mips64el http://pools.uniontech.com/desktop-professional eagle/sp3 main
contrib non-free
deb-mips64el http://pools.uniontech.com/desktop-professional eagle/sp3
main/debian-installer
deb-sw_64 http://pools.uniontech.com/desktop-professional eagle/sp3 main
contrib non-free
deb-sw_64 http://pools.uniontech.com/desktop-professional eagle/sp3
main/debian-installer
```

```
deb-src http://pools.uniontech.com/desktop-professional eagle/sp3 main contrib
non-free
clean http://pools.uniontech.com/desktop-professional
```

- 同步：`sudo apt-mirror`

更新上游仓库到主仓库

- 更新上游仓库到主仓库stable分支

  > 修改distributions中的update字段为对应的上游仓库的配置名称,比如1000, 1010

  ```
  pushd /var/www/repos/stable/device/ >/dev/null || exit
       sudo GNUPGHOME=/home/"$TUSER"/.gnupg reprepro -V update "$CODENAME"
  popd >/dev/null || exit
  ```

- 更新主仓库unstable分支到stable分支

  > 修改distributions中的update字段为对应的unstable仓库的配置名称，比如venus, mars/1010

  更新命令同上。

推送内网主仓库stable分支到外网仓库

- 同步脚本：`/data/script/rsync.sh`：

```
#!/usr/bin/bash
exec 1>>rsync.log 2>&1
date
rsync -avzP --delete --password-file=/etc/rsync.pass --include "dists/" --
include "pool/" --include "dists" --include "pool" --exclude "/*"
/data/repos/stable/device/ "chengdu@<ipaddr>::mirrors-ChengDu-device-repo"
```

- 添加定时任务`/etc/crontab`：

```
0 22    * * *   root    cd /data/script/ && ./rsync.sh
```

仓库的使用

- 添加仓库地址到 `/etc/apt/sources.list`

unstable仓库：`deb http://10.8.0.113/unstable/device/ CODENAME main contrib non-free`

stable仓库：`deb http://10.8.0.113/stable/device/ CODENAME main contrib non-free`

- 添加 软件源仓库公钥key

```
wget -O - http://10.8.0.113/unstable/devicepackages.key | sudo apt-key add -
sudo apt update
```

> 专用设备操作系统镜像已安装deepin-keyring，其已集成此key，无需再次添加。

- 修改仓库优先级

慎用混合仓库，如果你知道自己在做什么，添加了多个仓库，可能需要设置一下优先级：

`vi /etc/apt/preferences`

```
Package: *
Pin: origin 10.8.0.113
Pin-Priority: 900
```

> 数字越大，优先级越高，如果为-1,则禁用。

自建APT仓库管理脚本

为了方便仓库的搭建与管理，编写成了脚本，代码地址：Gitlab

Setup_Reprepro.sh

目标：在`/var/www/repos/apt`目录下自建指定创建多个 dist,例如`stable、unstable`，可以指定创建多个`repos`,例如`device`；可以指定多个 codename，例如`mars mars/1010  venus venus/1010`。

用法：普通用户执行命令`bash Setup_Reprepro.sh`

其中已设定：

```
GPGNAME=devicepackages
GPGEMAIL=devicepackages@uniontech.com
```

根据提示以此输入：

- dist:stable unstable and so on
- repos:device and so on
- codename: mars mars/1010 venus venus/1010 and so on

项目目录的.gnupg 会复制到主目录，如果项目目录没有会创建新的 gpg key。

如果没有/var/www/repos文件夹，将提示是否创建链接到/var/www/repos。

Add_to_APT_Repository.sh

目标：添加 crp 仓库软件包和源码到自建软件仓库。

用法：在 gitlab 更新软件代码后，在 crp 构建软件包之后，以普通用户执行命令

`bash Add_to_APT_Repository.sh dist repo codename crp_rep_url`

其中已设定：

```
dist: stable unstable
repo: device and so on
codename: mars mars/1010 venus venus/1010 and so on
crp_rep_url:crp_rep_url or local dir path
```

需要说明的是，输入或粘贴`crp_rep_url`后要跟上`/`以明确表示是目录，也可以跟本地目录。
注意此地址要和 repos 保持一致，不要将 device-cli 的 crp 仓库地址加到 device-gui 仓库，反之亦然。
专用设备 codename 有变化，device-gui 版本对应的 codename 为 mars，device-cli 版本对应的 codename 为 venus，故仓库 repos 统一为 device，只在 codename 中区分

从更新 gitlab 不同分支，到 crp 对应不同仓库构建软件包，到添加到 apt 仓库，需要人工分辨对应的是什么分支、什么版本、什么仓库，此部分操作需谨慎进行。

Man_APT_Repository.sh

目标：列出仓库软件包或者删除仓库中的软件包

用法：普通用户执行命令`bash Man_APT_Repository.sh dist repo codename action packagename`

其中已设定：

```
dist: stable unstable
repo: device and so on
codename: mars mars/1010 venus venus/1010 and so on
action: list remove
#list后不跟packagename
#remove支持一次性删除多个软件包,以空格间隔
```

> 删除软件包将同时删除源码包。

list/addtolist.sh

目标：增加软件仓库地址，比如 crp 地址，上游软件源地址，将末级目录添加到对应的软件更新清单。
用法：普通用户执行命令`bash addtolist.sh dist_repo_codename_comps.list`
其中已设定：

```
dist: unstable
repo: device and so on
codename: mars mars/1010 venus venus/1010 and so on
# codename中的"/"请转为"-"
comps: main contrib non-free
```

updatepackages.sh

目标：将 list/dist_repo_codename_comps.list 中定义的软件包添加到对应的仓库
用法：普通用户执行命令`bash updatepackages.sh [copy] [all]`

- 可选项：copy会执行软件包的copy操作，比如unstable_device_mars_main_copy_venus.list清单
- 可选项：copy all会遍历执行软件包的copy操作，比如unstable_device_mars_main_copy_all.list清单。
  默认已设置定时任务，配置在/etc/crontab 已定义，3 个小时检查一次更新。
  检测对比文件及日志在~/.cache/apt-repos/目录，如需强制更新，可删除此目录。

sync_base_and_unstable_to_stable.sh

目标：同步上游仓库。主要用途为主仓库，也就是 stable 仓库更新 base 仓库以及推送测试无误的 unstable 仓库到 stable 仓库。
用法：普通用户执行命令 `bash syncupstream.sh codename syncbase|syncdevice|syncall|checkbase|checkdevice|checkall [force]`

其中已设定：

```
codename: mars mars/1010 venus venus/1010 and so on
checkbase|checkdevice|checkall:检查更新base仓库、检查unstable仓库、检查全部
syncbase|syncdevice|syncall: 更新base仓库、更新unstable仓库、更新全部
force: 可选参数，强制更新
```

cache_packages_from_main_repos.sh

目标：通过chroot方式，与运行代码主机架构相同，从桌面版仓库抓包到本地，并添加到仓库。
用法：普通用户执行命令 `bash cache_packages_from_main_repos.sh`

设定：

```
非Amd架构，将download包后同步到服务器，同步后请到服务器添加包到仓库。

cache_packages_from_main_repos.sh 代码本身已设定为添加包时用copy方式，同时添加到mars和venus仓库。

list/fou-sp2/ list/eagle-sp2两个文件夹下有相应的抓包源和软件包列表。

按照规划，后期不再用抓包的方式，而是通过crp构建，故代码只设计了sp2仓库抓包。
**此代码已停用**
```

vsftp.sh

目标：为apt软件源仓库启用ftp协议支持。
用法：普通用户执行命令 `bash vsftp.sh`

webtree

目标：辅助网络地址软件包的添加。
功能：获取软件包地址，添加软件时会调用。

参考：

1. SetupWithReprepro
2. Creat your own apt repo

文档信息

- 发表日期：2020-12-24

---

1. https://www.debian.org/doc/manuals/debian-reference/ch02.zh-cn.html "Debian软件包管理" ↩
2. https://www.ruanyifeng.com/blog/2013/07/gpg.html "GPG入门教程" ↩

3. https://blog.packagecloud.io/eng/2014/10/28/howto-gpg-sign-verify-deb-packages-apt-repositories/ "GPG sign and verify deb" ↩
4. https://www.yiibai.com/apache_http "Apache教程" ↩
5. https://wiki.archlinux.org/index.php/Very_Secure_FTP_Daemon_(%E7%AE%80%E4%BD%93%E4%B8%AD%E6%96%87) "VSFTP" ↩
6. https://salsa.debian.org/brlink/reprepro "Reprepro" ↩