apt-get

apt VS apt-get

- https://itsfoss.com/apt-vs-apt-get-difference/
- https://itsfoss.com/apt-get-linux-guide/
- https://itsfoss.com/apt-command-guide/

sources.list

- copia: **/usr/share/doc/apt/examples/sources.list**
- añadir **universe** y **multiverse**:

```
sudo add-apt-repository universe
sudo add-apt-repository multiverse
```

ignore certificate

Cuando estamos en una versión vieja de algún linux (debian 9 strech en este caso), podemos intentar encontrar la manera de actualizar los certificados o simplemente ignorarlo)

Estamos hablando de ignorar que que no se puede verificar la validez del certificado (ataques man-in-the-midle), porque no diponemos de los certificados raiz del que firma ese certificado.

añadir (no me funcionó)

- https://stackoverflow.com/questions/21181231/server-certificate-verification-failed-cafile-etc-ssl-certs-ca-certificates-c/67698986#67698986
- miro quien ha firmado el certificado:

```
echo -n | openssl s_client -showcerts -servername packages.sury.org -connect packages.sury.org:443 2>/dev/null | tac | awk '/-END CERTIFICATE-/{f=1} f;/-BEGIN CERTIFICATE-/{exit}' | tac | openssl x509 -noout -subject -issuer
```

  - respuesta:

```
subject=C = US, O = Internet Security Research Group, CN = ISRG Root X1
issuer=O = Digital Signature Trust Co., CN = DST Root CA X3
```

- descargo el certificado raiz de Let's Encrypt (https://letsencrypt.org/certificates/)
- verifico e instalo:

```
openssl x509 -noout -text -in isrgrootx1.pem
sudo mv isrgrootx1.pem /usr/local/share/ca-certificates/isrgrootx1.crt
sudo update-ca-certificates
```

  - se supone que añade a **/etc/ssl/certs/ca-certificates.crt** ?

ignorar

- https://unix.stackexchange.com/questions/317695/is-it-possible-to-have-apt-accept-an-invalid-certificate
- ignora el **server certificate verification failed. CAfile: /etc/ssl/certs/ca-certificates.crt CRLfile: none**:

  /etc/apt/apt.conf.d/80ssl-exceptions.conf

  ```
  Acquire::https::packages.sury.org::Verify-Peer "false";
  Acquire::https::packages.sury.org::Verify-Host "false";
  ```

- solución temporal si ficheros de configuración:

```
apt -o "Acquire::https::Verify-Peer=false" update
```

remove app

- ver paquetes instalados: `apt list --installed`
- https://askubuntu.com/questions/187888/what-is-the-correct-way-to-completely-remove-an-application
- `remove` : elimina binarios pero no ficheros de configuración
- `purge` o `remove --purge` : lo elimnina todo menos las dependencias
  - interesante para empezar de cero
- `autoremove` : elimina paquetes huérfanos que estaban instalados como dependencia de otros
- `aptitude remove` o `aptitude purge` : también elimina paquetes no ya requeridos (a menos que los necesite otro paquete). **aptitude** solo recuerda la información de dependencia de los paquetes que han sido instalados desde **aptitude**
- `sudo apt-get remove «^aplicaciont.*`
- para averiguar a que paquete pertenece un fichero: `dpkg -S /path/al/fichero`

remove unused

- `apt-get clean` : eliminar paquetes .deb ya instalados
- `apt-get autoclean` : eliminar paquetes que ya no existen en el repositorio o que tienen versiones posteriores
- `apt-get autoremove` : eliminar paquetes vinculados a otros y que no son necesarios
- `apt-get remove –purge linux-image-X.X.XX-XX-generic` : eliminar versiones anteriores de kernel
  - `dpkg --get-selections | grep linux-image`

apt-mark

marcar paquetes que no serán actualizados.

- `apt-mark hold <paquete> <paquete>…`
- `apt-mark unhold <paquete>`
- `apt-mark`
  - **auto**: marca los paquetes como instalables automaticamente. Si ningún otro paquete depende de él, se desinstala (como la satisfacción de dependencias cuando instalas un paquete)
  - **hold**: paquete retenido

    - **manual**: marca los paquetes como instalables manualmente. Aunque no haya paquetes que no dependan de él, no será desinstalado automaticamente.q
    - **showauto**: lista de paquetes marcados como automaticos
    - **showhold**: lista de paquetes marcados como retenidos
    - **showmanual**: lista de paquetes marcados como manual
    - **unhold**: desmarca paquete retenido
- ansible:

```
# Prevent nginx from being upgraded
- dpkg_selections:
    name: python
    selection: hold

# Kept multiple packages back (hold packages)
- dpkg_selections: name={{ item }} selection=hold
  with_items:
          - apache2
          - php7-fpm
          - nginx
          - mariadb-server

# Removing hold using Ansible
- dpkg_selections:
    name: python
    selection: install
```

/via: https://www.cyberciti.biz/faq/apt-get-hold-back-packages-command/

apt-file

- equivalente a `yum provides`
- `apt install apt-file`
- `apt-file update`
- `apt-file search <file>`

/via: https://sysadmincasts.com/episodes/41-cli-monday-apt-file-and-yum-provides

comandos

- `apt-get update` : actualiza la lista de paquetes de los repositorios indicados en sources.list. Es lo primero que se debe hacer antes de instalar nada.
- `apt-get install <paquete>` : una vez conocemos el paquete que queremos instalar, lo indicamos y a descargar. APT-GET nos informará de que se descarga, que se actualiza, que otros paquetes se instalan relacionados con este, la cantidad de espacio que ocupará, etc...
- `apt-get upgrade`: actualiza los paquetes de soft instalados en el sistema.
- `apt-cache search <cadena>` : sirve para buscar la <cadena> en los repositorios y ver si encontramos el paquete que coincida. A veces ayuda si ya sabes el nombre del paquete ;)
- `apt-cache policy` : comprueba el estado de los repositorios existentes
- `apt list --installed | grep <CADENA>`
- `apt-get dist-upgrade` : actualiza paquetes de la distribución.
- `apt-get remove <paquete>` : elimina el paquete del sistema
- `apt-get clean` : Limpiar cache de aplicaciones instaladas

- `apt-get autoclean` : Limpiar aplicaciones no instaladas
- `apt-get autoremove` : Limpiar posibles dependencias de aplicaciones desinstaladas
- `apt-get install <paquete> -d` : --download-only --no-download
- `apt-get download <paquete>`
- `apt-get install <paquete> -s` : --simulate
- `apt-get install --reinstall <paquete>`
- `apt-file search <fichero>` : busca en que paquete está un determinado archivo
- `apt-file list <paquete>` : lista los ficheros contenidos en un paquete

apt

- `apt list –installed | grep libc6`

versiones

se puede mirar que versión específica de un paquete hay disponible e instalarla:

```
apt-cache madison <paquete>
```

```
apt-get install <paquete>=<version>
```

otros

- herramienta que chequea todas las dependencias de un paquete : `$ apt-rdepends <paquete>`
  - se puede instalar directamente desde repositorio: `$ apt-get install apt-rdepends`
- otros repositorios
- `apt-file`: busca que paquete tiene un fichero en concreto (búsqueda de comandos)
  - `sudo apt-file update`
  - https://sysadmincasts.com/episodes/41-cli-monday-apt-file-and-yum-provides

casos de uso

update kernel

- `apt-cache search linux-image`
- `sudo apt-get install <linux-image-flavour>`
- forzar el repositorio que queremos usar: `sudo apt-get install -t wheezy-backports linux-image-amd64`

caso de uso, comandos ejemplo

- `apt-cache show linux-image-amd64`
- `apt-cache policy linux-image-amd64`

/via: https://serverfault.com/questions/670088/install-debian-backports-kernel-automatically

crear paquete .deb

para crear un paquete en el que incluir el software que queramos distribuir (por la razón que sea) hemos de:

1. crear una estructura de directorios determinada (ver más adelante)
2. ejecutar

```
dpkg-deb -b <PATH>/ .
```

estructura directorio

```
.── DEBIAN
¦   ¦── control
¦── <ubicación archivos en root sistema>
¦── <ubicación archivos en root sistema>
```

control

```
Package: ibscanultimate
Version: 3.0.0-1
Maintainer: You <whatever@contact.address>
Architecture: amd64
Description: IBScanUltimate
 Software for the integrated biomedics kojak scanner
```

ejemplo:

construccio_manual_paquet_debian.tar.gz

trucos

solucionar bloqueo "lock"

`E: Could not get lock /var/lib/apt/lists/lock - open (11: Resource temporarily unavailable)` → `sudo rm /var/lib/apt/lists/lock`

proxy debian 10 buster

/etc/environment

```
http_proxy=ipserverproxy:Puerto
https_proxy=ipserverproxy:Puerto
ftp_proxy=ipserverproxy:Puerto
# con autentificación
http_proxy=user:password@ipserverproxy:Puerto
https_proxy=user:password@ipserverproxy:Puerto
ftp_proxy=user:password@ipserverproxy:Puerto
```

/etc/apt/apt.conf.d/02proxy

```
Acquire {
  HTTP::proxy "http://PROXYSERVERIP:PROXYPORT";
  HTTPS::proxy "http://PROXYSERVERIP:PROXYPORT";
}

# con autentificación (sin agrupar es opcional)
Acquire::http::Proxy «http://usuario:password@ipserverproxy:puerto»;
Acquire::ftp::Proxy «ftp://usuario:password@ipserverproxy:puerto»;
```

/via:

- https://kifarunix.com/configure-apt-proxy-on-debian-10-buster/
- https://www.sugeek.co/configurar-proxy-en-debian-por-consola/

proxy con autentificación

editar con este comando: `sudo gedit /etc/apt/apt.conf` o si hemos hecho el paso ubuntu 11.10 hacer solo esto: `sudo cp /etc/apt/apt.conf.d/02proxy /etc/apt/apt.conf`

```
<code>
Acquire::http::Proxy “http://usuario:contraseña@proxy:puerto″;
Acquire::https::Proxy “http://usuario:contraseña@proxy:puerto″;
Acquire::ftp::Proxy “http://usuario:contraseña@proxy:puerto″;
Acquire::socks::Proxy “http://usuario:contraseña@proxy:puerto″;
```

fuente: http://www.linuxquestions.org/questions/ubuntu-63/problems-with-apt-get-synaptic-and-proxy-454026/

proxy con autentificación (a partir de ubuntu 11.10?)

editar con este comando: `sudo gedit /etc/apt/apt.conf.d/02proxy` añadir:

```
Acquire::http::Proxy “http://usuario:contraseña@proxy:puerto″;
Acquire::https::Proxy “http://usuario:contraseña@proxy:puerto″;
Acquire::ftp::Proxy “http://usuario:contraseña@proxy:puerto″;
Acquire::socks::Proxy “http://usuario:contraseña@proxy:puerto″;
```

fuente: http://naveenubuntu.blogspot.com.es/2011/09/updating-packages-behind-prxy-in-ubuntu.html