Arch Linux Packge 的前世今生

Chih-Hsuan Yen

(yan12125)

COSCUP x GNOME.Asia x openSUSE.Asia 2018

COSCUP 2018

archlinux.tw

About Me

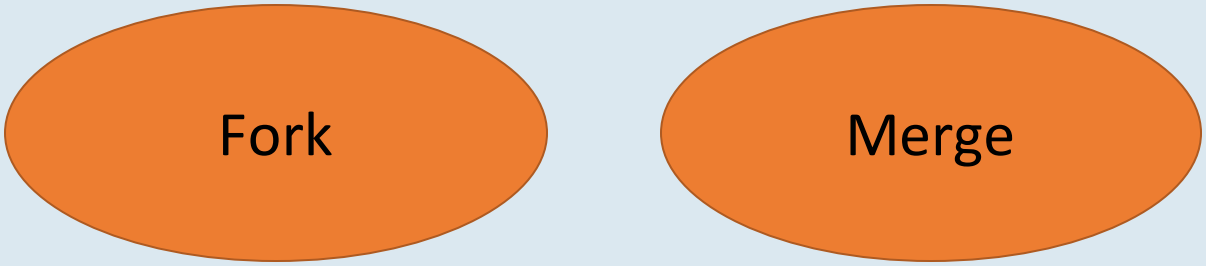
- A PhD student from National Taiwan University
  - Embedded systems, deep learning, storage
- 打包狗
  - AUR & ArchlinuxCN
  - MacPorts
  - python3-android

(台灣犬)

- Work as hard as a dog
- As tired as a dog

What are packages?

- The basic unit of a distribution
- Why packaging
  - Learn software internals
  - An expression of freedom



“凝聚共識的過程” - 唐鳳, 2018

Philosophy

- UNIX philosophy – do simple tools well
- Arch Linux philosophy: KISS – Keep It Simple & Stupid
  - Simple architecture
  - Simple file formats

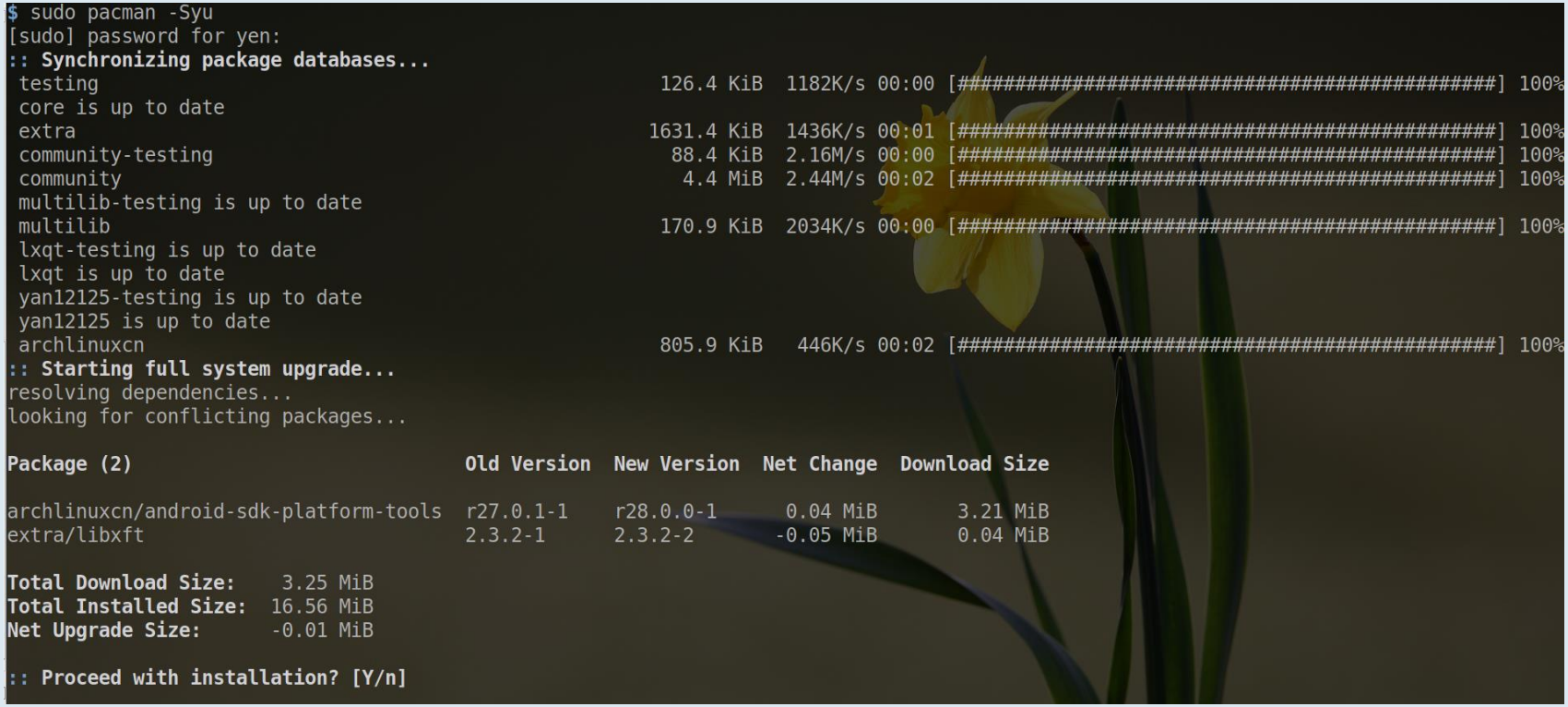
始臣之解牛之時，所見無非全牛者

道

三年之后，未嘗見全牛也

方今之時，臣以神遇而不以目視，官知止而神欲行

Daily routines for Arch Linux users

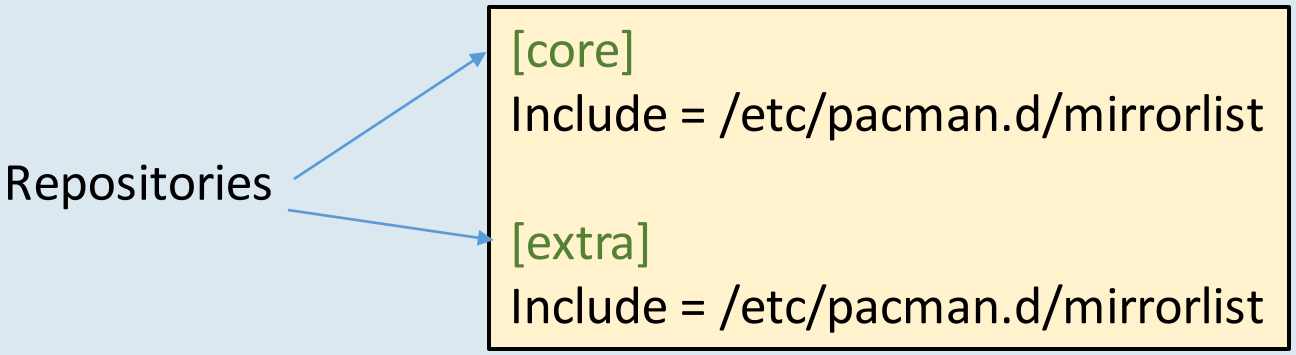


COSCUP 2018

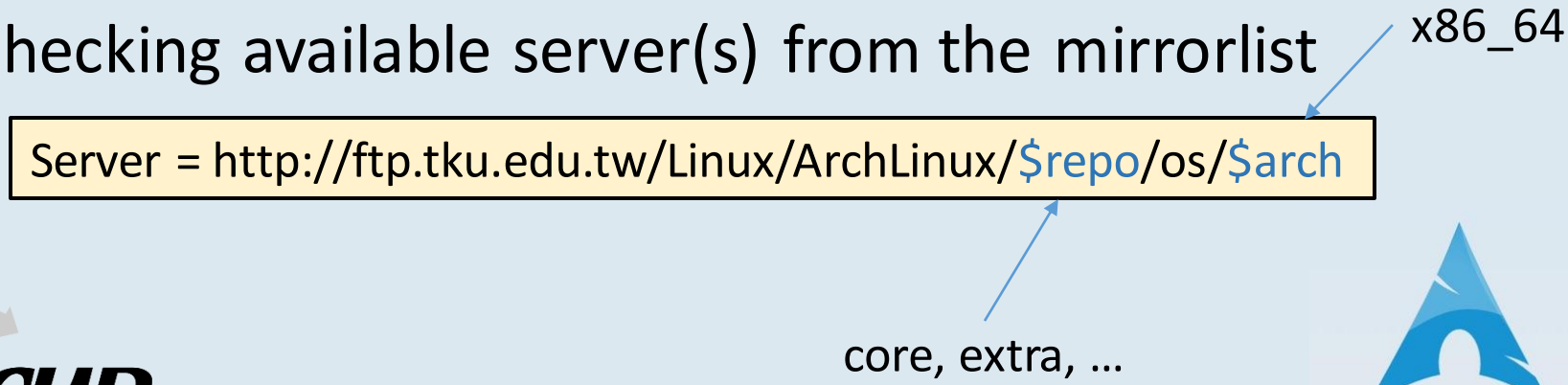
archlinux.tw

What's behind the scenes?

- Listing repositories from `/etc/pacman.conf`

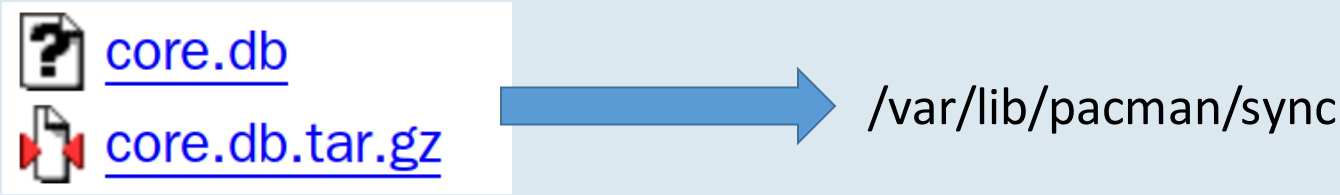


- Checking available server(s) from the mirrorlist



What packages do you have?

- Downloading package databases from mirrors



- Analyzing repo databases
  - Each `foo.db` is a gzipped tarball
  - Each folder in this tarball describes a package

```
$ tar tf /var/lib/pacman/sync/core.db
acl-2.2.52-4/
acl-2.2.52-4/desc
archlinux-keyring-20180404-1/
archlinux-keyring-20180404-1/desc
```

What packages do you have?

- Package description file

```
$ tar -xOf /var/lib/pacman/sync/core.db archlinux-keyring-20180404-1/desc
%FILENAME%
archlinux-keyring-20180404-1-any.pkg.tar.xz

%NAME%
archlinux-keyring

%VERSION%
20180404-1

%DESC%
Arch Linux PGP keyring

%CSIZE%
684236

%ISIZE%
948224
```

Installing packages

- Downloading package(s)

archlinux-keyring-20180404-1-any.pkg.tar.xz → /var/cache/pacman/pkg

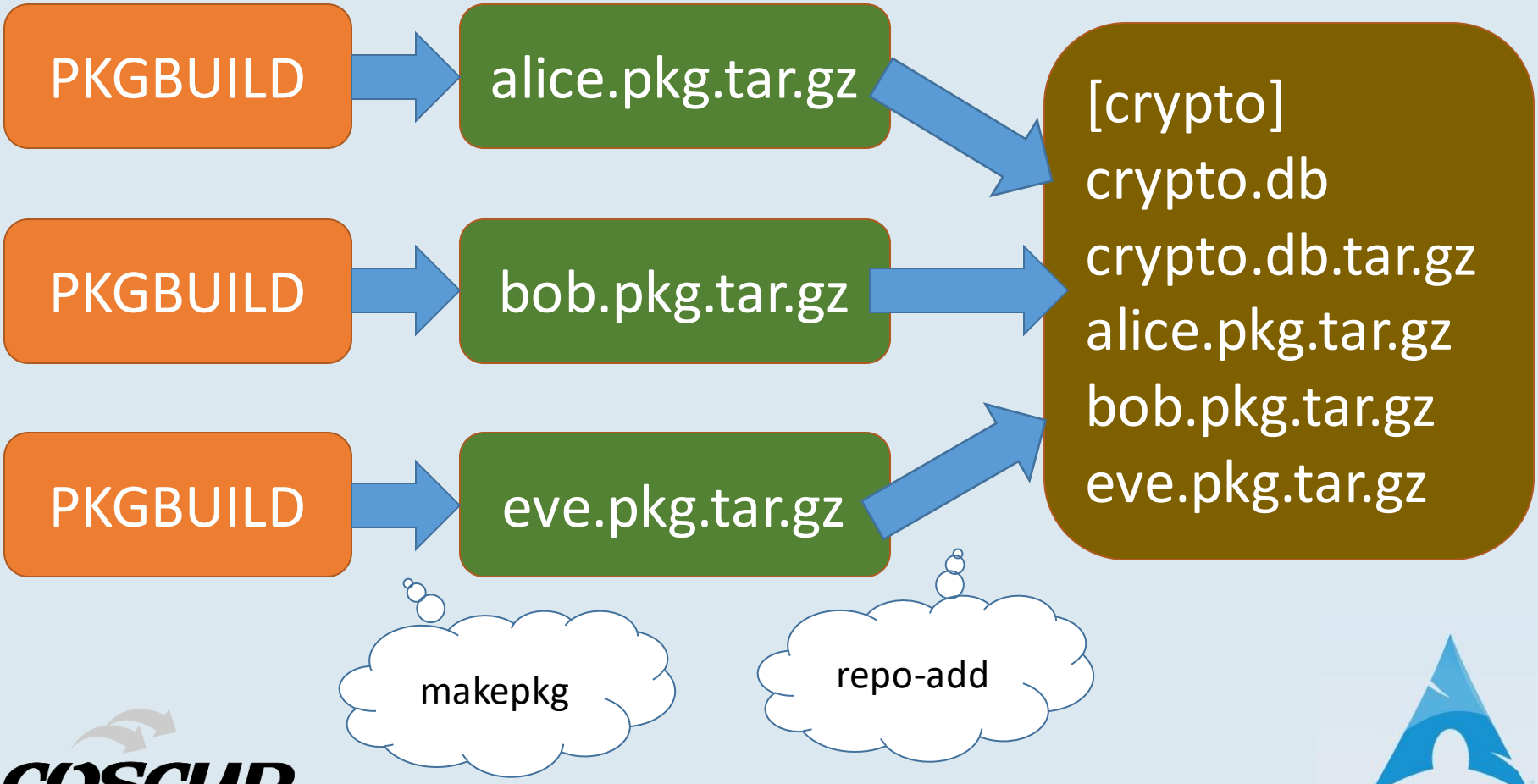
- Packages are xz-compressed tarballs

```
$ tar tf /var/cache/pacman/pkg/archlinux-keyring-20180404-1-any.pkg.tar.xz
.PKGINFO
.BUILDINFO
.INSTALL
.MTREE
usr/
usr/share/
usr/share/pacman/
usr/share/pacman/keyrings/
usr/share/pacman/keyrings/archlinux.gpg
usr/share/pacman/keyrings/archlinux-trusted
usr/share/pacman/keyrings/archlinux-revoked
```

Package metadata files

- *.PKGINFO*: general package info
  - Related to installation: package size, signature, dependencies, conflicts
- *.BUILDINFO*: new in pacman 5.1, for reproducible builds
- *.MTREE*: gzipped mtree file
  - Hash checksums, size, permission, modification time for each file in the package
- *.INSTALL*: the install script

Generation of packages and repos



PKGBUILD

- A shell script
- Structure
  - Basic information – pkgname, pkgver, source
  - prepare()
  - pkgver()
  - build()
  - check()
  - package()
- Other files: patches, data files, install script

A sample PKGBUILD - which

```
pkgname=which
pkgver=2.21
pkgrel=2
pkgdesc='A utility to show the full path of commands'
arch=('x86_64')
url='https://savannah.gnu.org/projects/which/'
license=('GPL3')
groups=('base' 'base-devel')
depends=('glibc' 'bash')
source=("https://ftp.gnu.org/gnu/$pkgname/$pkgname-$pkgver.tar.gz")
md5sums=('097ff1a324ae02e0a3b0369f07a7544a')
```

A sample PKGBUILD - which

```
build() {
  cd $pkgname-$pkgver
  ./configure --prefix=/usr
  make
}

package() {
  cd $pkgname-$pkgver
  make DESTDIR="$pkgdir" install
}
```

Once upon a time…

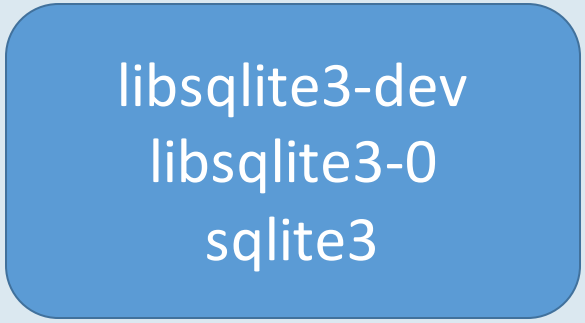
- There's only one *build()* function, and `makepkg` is run as root
  - Package files should not be owned by the user running `makepkg`
- Later fakeroot is introduced
- 2009/01: *package()* is added (v3.2.2-61-g08034ceb)
  - Matching this pattern: make && sudo make install
  - "Minimize fakeroot usage"

History of PKGBUILD

- 2010/12: *check()* is added (v3.4.2-203-g0c29eb43)
- 2012/08: *prepare()* is added (v4.0.3-326-g065b7f86)
  - Getting VCS sources done
  - Applying patches
  - `autoreconf`
- 2012/08: *pkgver()* is added (v4.0.3-352-g888020de)
  - For updating pkgver of VCS packages automatically

Arch Linux packaging practices

- Keep close to upstream
- Avoid bundled sources
- No development packages in general (不拆包)

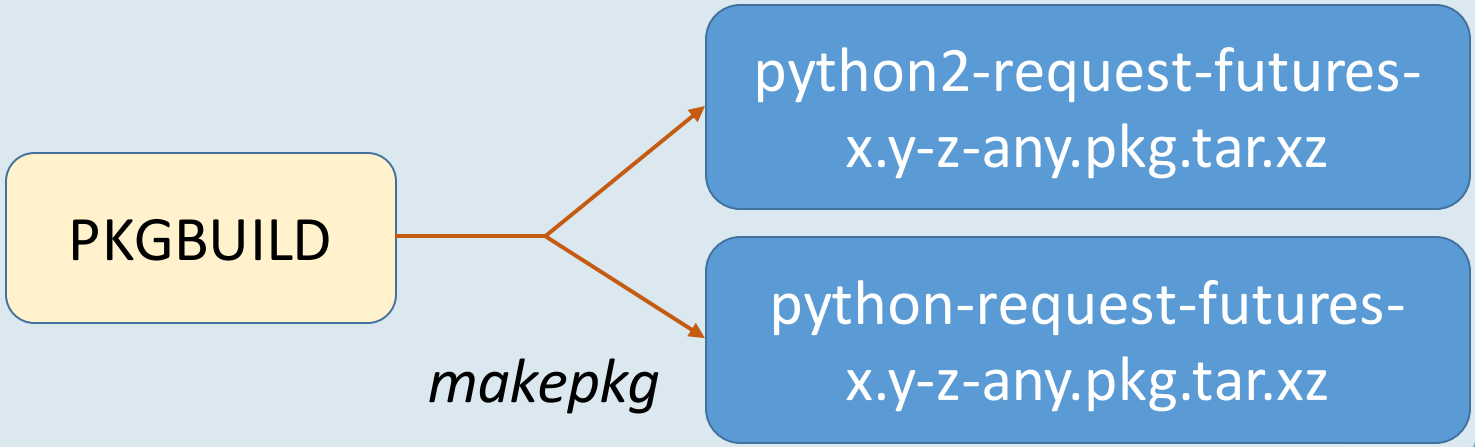


(Debian)

(Arch Linux)

Split packages (v3.2.2-67-g3d49d880)

- Some packages (e.g. Python 2/3 variants of a Python package) share some packages information and/or build steps
- Generating multiple packages from one PKGBUILD

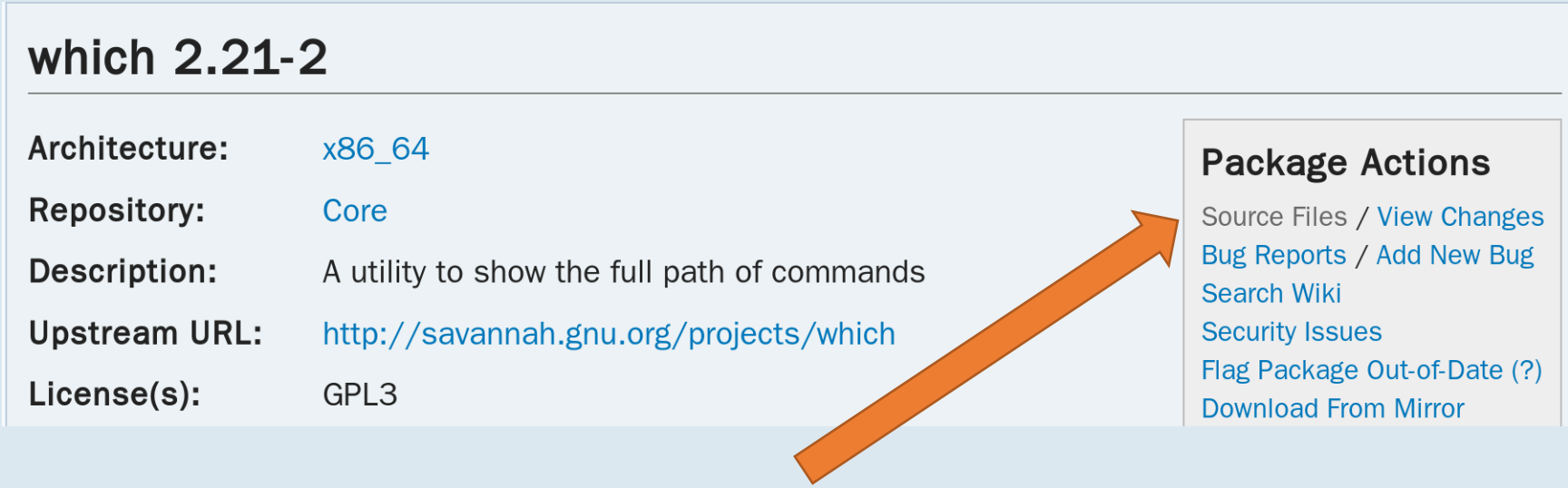


Split packages

```
package_python-requests-futures() {
    depends=('python-requests')
    cd $_distname
    python setup.py install --root="${pkgdir}" --optimize=1
}
package_python2-requests-futures() {
    depends=('python2-requests' 'python2-futures')
    cd "$_distname-py2"
    python2 setup.py install --root="${pkgdir}" --optimize=1
}
```

Learning from PKGBUILDs

Arch Linux homepage -> Package search



Learning from PKGBUILDs

AUR homepage -> Package search

Sharing Packages

```
pkgname=which
pkgver=2.21
pkgrel=2
pkgdesc='...'
arch=('x86_64')
...
```

Distribute PKGBUILD

Distribute binary packages

Sharing PKGBUILDs

- Personal VCS repo
  - Many on GitHub
- AUR (Arch User Repository)

https://aur.archlinux.org/

Uploading to AUR

- AUR3
  - *mkaurball / makepkg --source* => foo.src.tar.gz
  - Login to AUR and upload the tarball via the web interface
- AUR4
  - Each package is a git repo
  - *makepkg --printsrcinfo > .SRCINFO*
  - *git push* to aur@aur.archlinux.org:foo.git

After uploading to AUR…

- phantomjs

> **boris220 commented on 2018-05-14 21:37**
>
> There is a make dependency missing in the PKGBUILD: bison is needed to build qtwebkit

- webkitgtk

> **germanfr commented on 2017-12-30 16:30**
>
> It took me 7h to build this package as an update. That's not ok.

Good AUR comments

- darling-dmg-git

jamesan commented on 2016-01-27 11:35

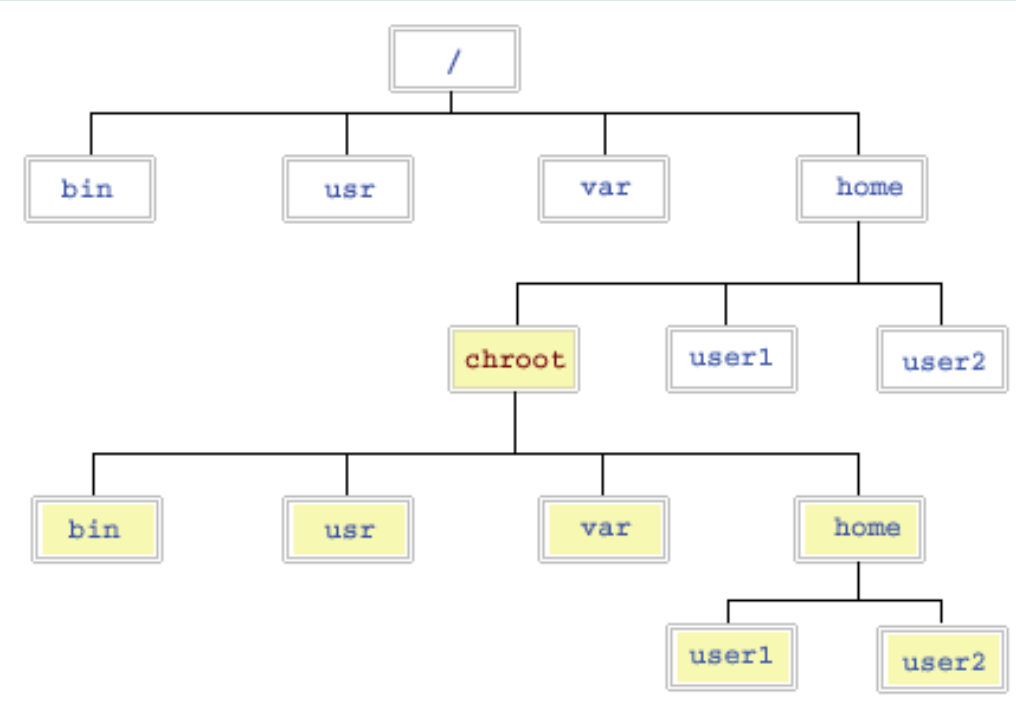
I flagged this as out-of-date as the last upstream version, 1.0.3, was released in Dec 2015. You can find the modified PKGBUILD here:

https://github.com/jamesan/darling-dmg-git/blob/master/PKGBUILD

I also modified significant portions of the PKGBUILD to bring it closer to current packaging guidelines:

added a check() function to run post-build/pre-install tests;
added a pkgver() function to derive the version string from the source; and,
removed redundant bits like including base group packages or using the $srcdir when the PWD is already set to $srcdir by makepkg.

Use the same environment for building



1

Install the “base-devel” group in a chroot

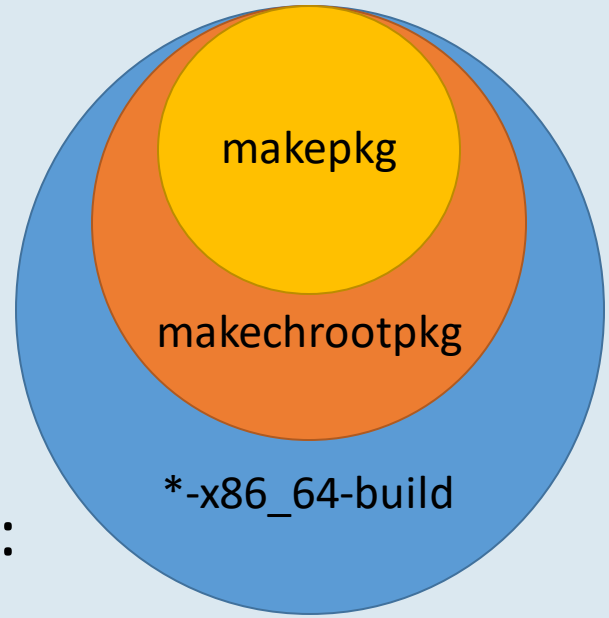
2

Build in the chroot


1
按下鼠标
2
拖动鼠标
3
释放鼠标
移动到 冰箱(B)
大象(A)
冰箱(B)
冰箱(B)

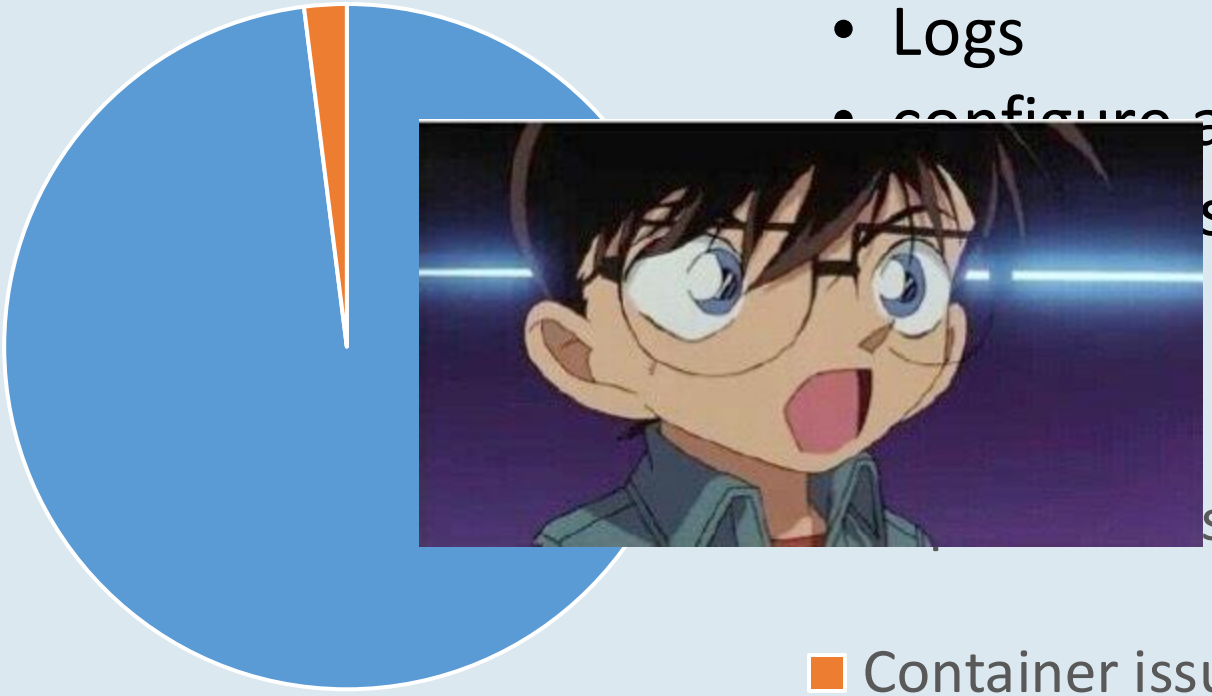
Chroot-related commands

- pacstrap: install packages to the given folder
- mkarchroot: create a chroot using pacstrap
- makechrootpkg: run makepkg in a chroot
- {extra,testing,staging}-x86_64-build: run all steps above



*** Need the "devtools" package ***

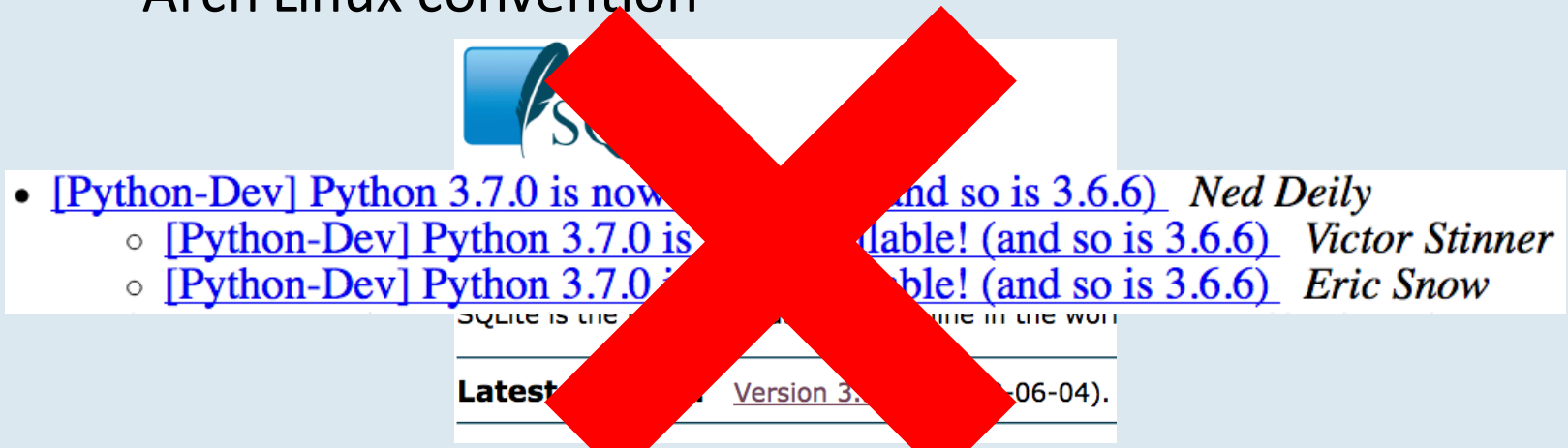
Common issues in chroots



- Logs
- configure.ac,
- s.txt, setup.py
- s

Keeping Packages Up-to-date

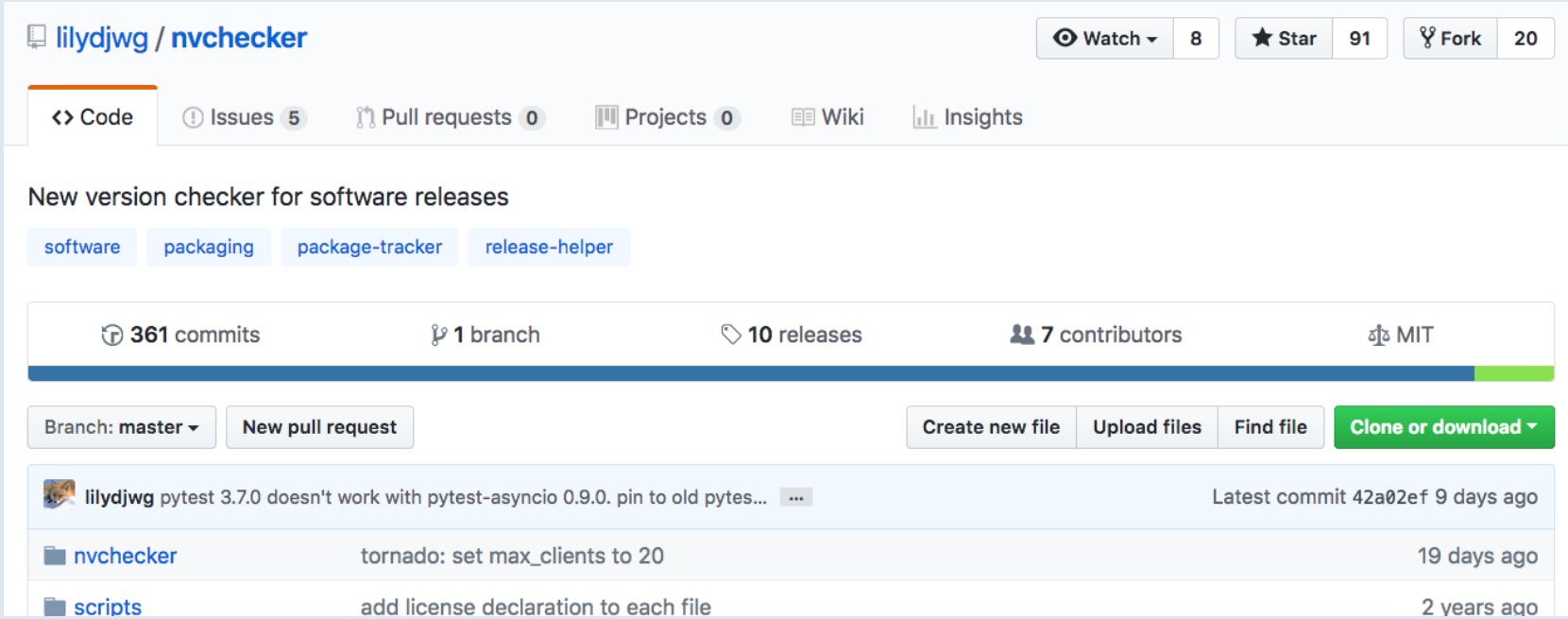
- Why keeping packages up-to-date?
  - Arch Linux convention



"We will encourage you to develop the three great virtues of a programmer: *laziness*, *impatience*, and *hubris*." -- LarryWall, Programming Perl

Keeping Packages Up-to-date Automatically

https://github.com/lilydjwg/nvchecker/



COSCUP 2018

archlinux.tw

nvchecker

- Checking multiple softwares using a configuration file, usually *nvchecker.ini*
- Independent of package managers
  - Use *oldver* file and *newver* file to record versions
- Multiple backends
  - AUR, PyPI, NPM, GitHub, regular expressions
- Parallel checking
- Need to run *nvtake* to update *old_ver.txt* for each software

An example nvchecker.ini

```
[__config__]
oldver = old_ver.txt
newver = new_ver.txt

[McBopomofo]
github = openvanilla/McBopomofo
use_latest_release = true

[py-jsbeautifier]
pypi = jsbeautifier

[FileZilla]
url = https://filezilla-project.org/versions.php?type=client
regex = <a name="([\d.]+)">
```

Running nvchecker

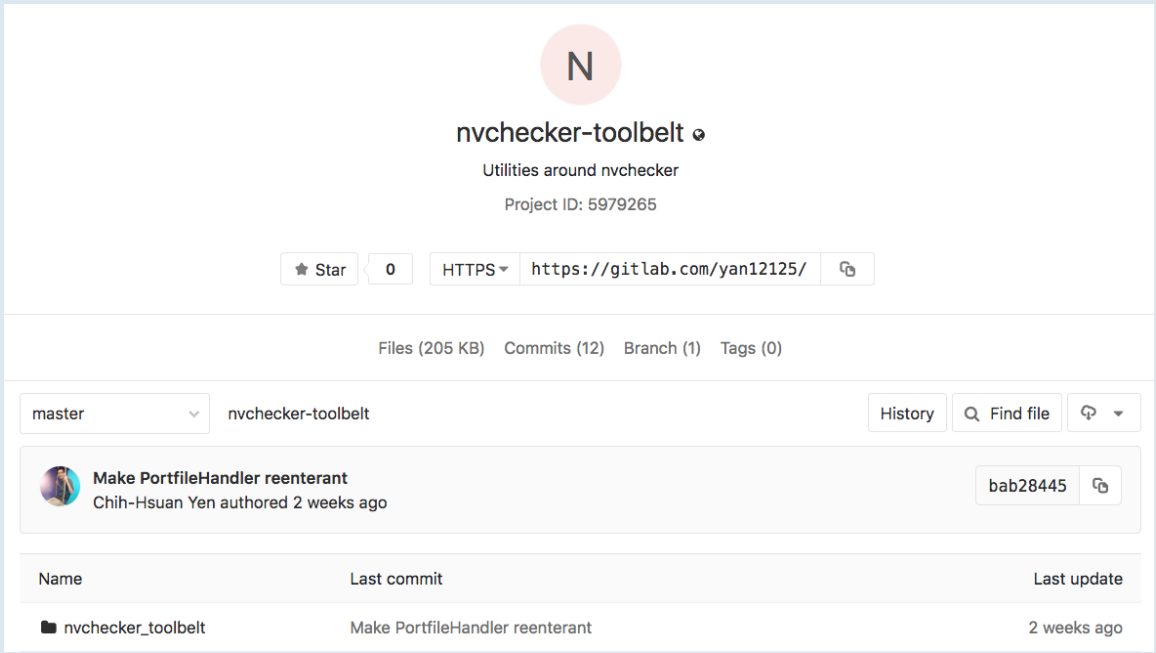
- *nvchecker nvchecker.ini*

```
[I 08-10 17:17:06.283 _base:136] gn-git: updated from 1.11.091712940324bb8cfb48340e00785118ee8814
1a to 1.12.c0744edfd85862677d625f0fdadda9edc95eed35
[I 08-10 17:17:06.508 _base:136] rr-git: updated from 1.482.0b54453e0c92c293defd1fe9694fe5bd05d60
313 to 1.483.05fc727843d45ff0ec18cefebda6b42d20446063
[I 08-10 17:17:06.703 _base:136] emacs-llvm-mode-git: updated from 1.328.36f54002c931a026f490f9fb
074c11d91e3487a2 to 1.329.252445ebb1e2785a5d6e1c733e975a84720659ce
[I 08-10 17:17:07.426 _base:136] gcsf-git: updated from 20180801.083401 to 20180810.082040
[I 08-10 17:17:07.434 _base:136] gimp-git: updated from 1.205.537bf4ec6a9c99894f4df7409c78b2fea73
ef381 to 1.206.29a05ccfacb9ce4a2fdaf6f048a06fed74fbf698
[I 08-10 17:17:08.073 _base:136] tidb-git: updated from 20180809.144524 to 20180810.081910
[I 08-10 17:17:09.888 _base:136] nheko-git: updated from 20180809.175108 to 20180810.075846
[I 08-10 17:17:10.086 _base:136] kodi-git: updated from 20180809.065354 to 20180810.082031
[I 08-10 17:17:10.182 _base:136] scummvm-git: updated from 20180809.223139 to 20180810.054744
[I 08-10 17:17:10.811 _base:136] supertuxkart-git: updated from 20180809.234755 to 20180810.08392
9
[I 08-10 17:17:10.934 _base:136] archrepo2-git: updated from 20180625.062836 to 20180810.085413
[I 08-10 17:17:10.946 _base:136] python-git: updated from 20180809.204949 to 20180810.060208
[I 08-10 17:17:11.026 _base:136] scrcpy: updated from 1.2-1 to 1.3-1
[I 08-10 17:17:11.531 _base:136] masterpdfeditor: updated from 5.1.00-1 to 5.1.12-1
[I 08-10 17:17:11.775 _base:136] anbox-git: updated from 20180808.092809 to 20180810.072434
[I 08-10 17:17:11.957 _base:136] dolphin-emu-git: updated from 20180809.085129 to 20180810.091557
```

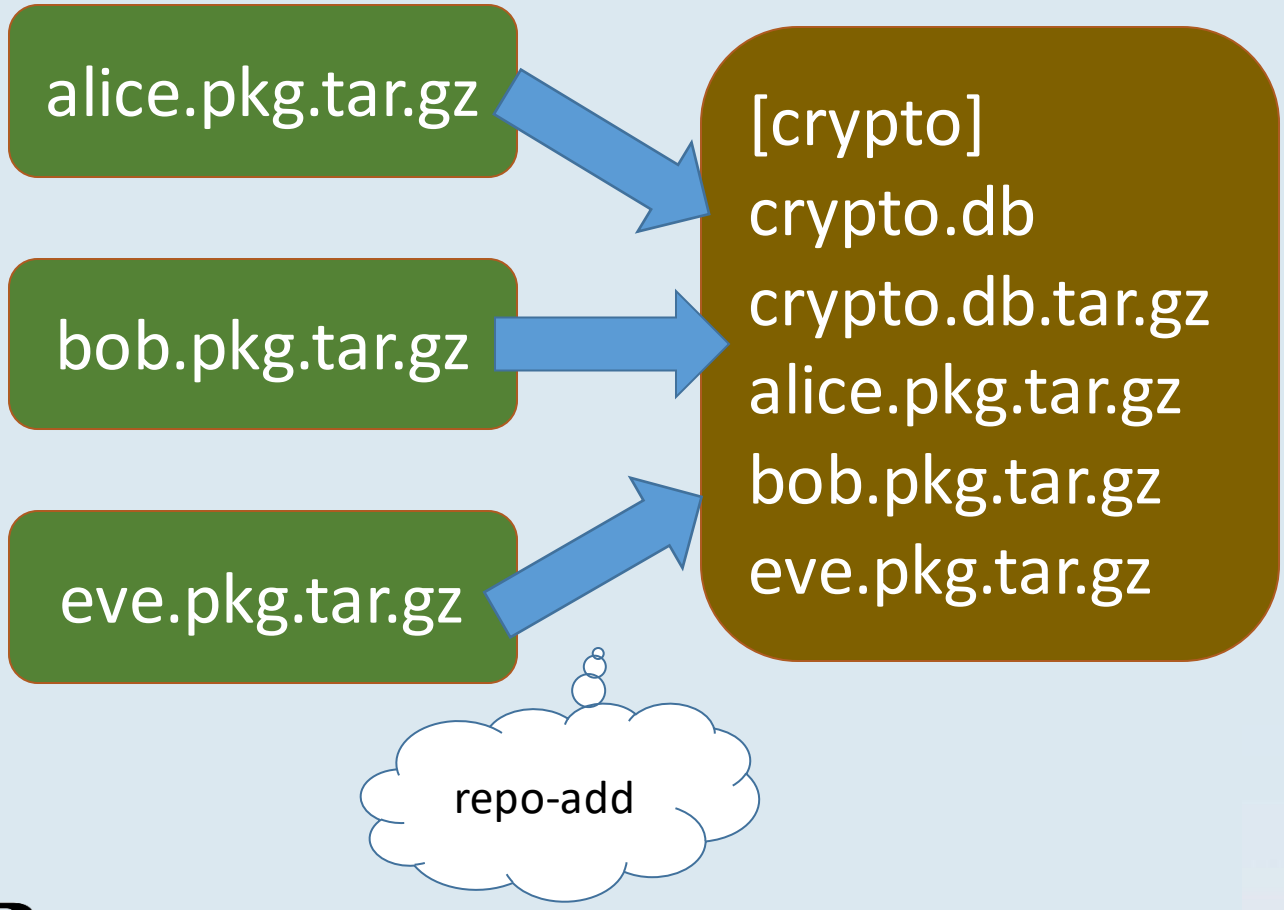
Integration with Package Managers

- Why not just update PKGBUILD?

  https://gitlab.com/yan12125/nvchecker-toolbelt/



Unofficial Repositories

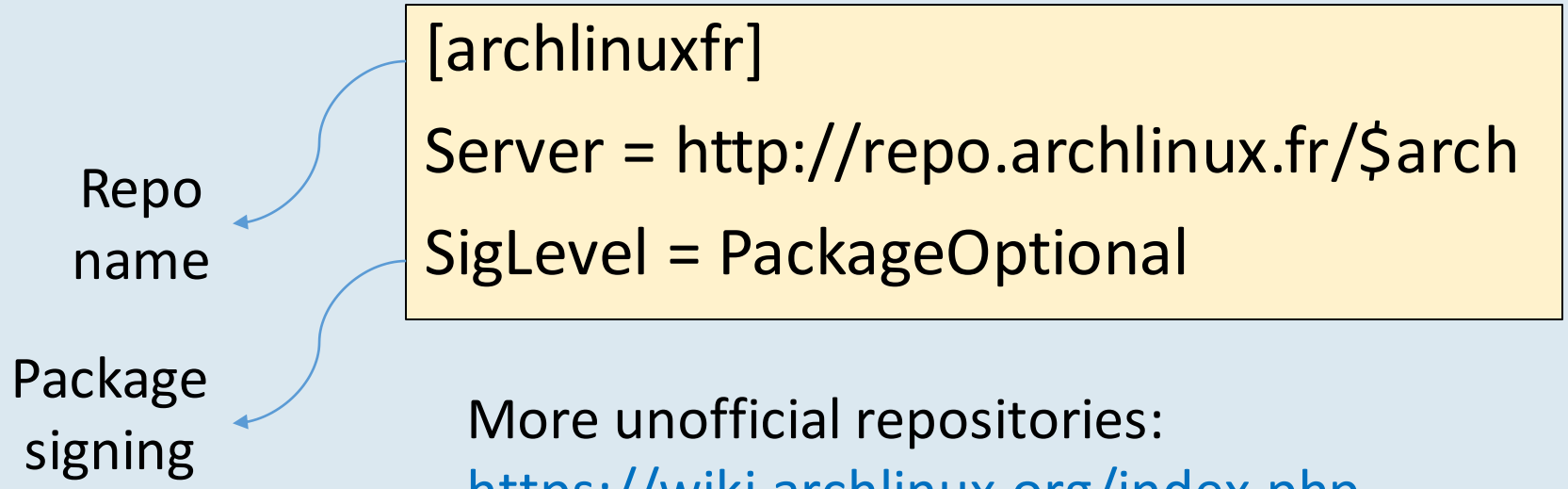


Creating Unofficial Repositories

- repo-add myrepo.db.tar.gz foo.pkg.tar.xz
- repo-remove myrepo.db.tar.gz foo
- repo-elephant

Using Unofficial Repositories

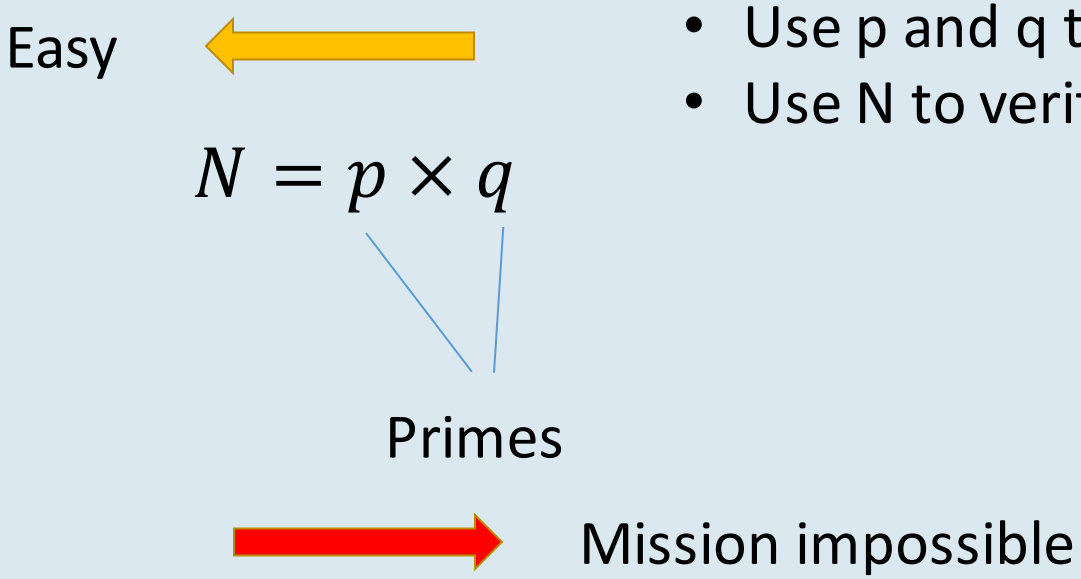
(Append to */etc/pacman.conf*)



More unofficial repositories:
https://wiki.archlinux.org/index.php/unofficial_user_repositories

How to trust binaries packages?

- A little cryptography (RSA)



- Use p and q to sign
- Use N to verify

- Use *makepkg --sign* to sign packages

Packaging rebuilds

- Rebuild a package without actually changing PKGBUILD
  - Only pkgrel is bumped
- Common scenarios
  - Changed SONAME (name of a shared library)

  - Changed runtime library paths (e.g., Python, Perl)

Staging repositories

- For storing rebuilt packages temporarily

/usr/bin/python3.7

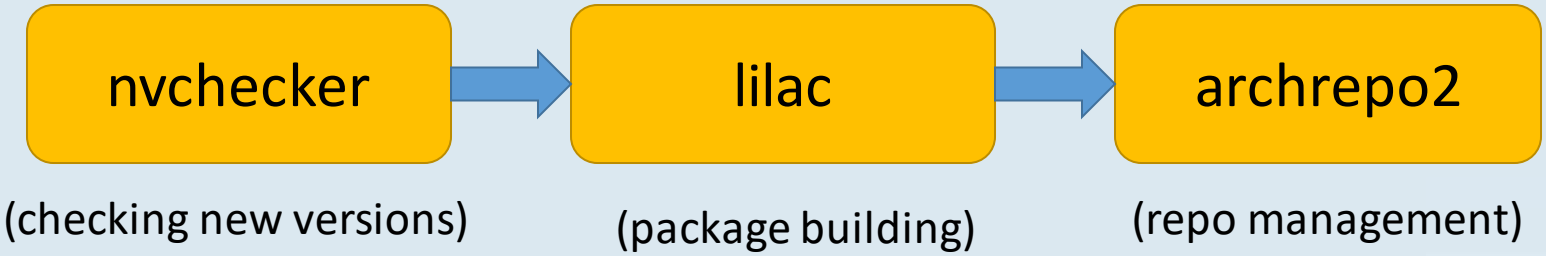
/usr/lib/python3.7/site-packages/bar

/usr/lib/python3.6/site-packages/foo ?

- Rebuild all packages to staging/community-staging/multilib-staging and move them all at once

Automatic packaging

- Manual packaging
  - Check software versions
  - Updating PKGBUILD
  - *extra-x86_64-build*
  - *repo-add*
- Automatic packaging



Arch Linux China community

- Maintains one of the largest unofficial repository [archlinuxcn]
- Develops several tools/scripts to make packaging managing easier

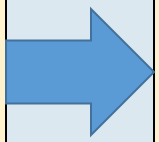
https://github.com/archlinuxcn/

Automatic Packaging with nvchecker & lilac

Case I: pull packages from AUR

```
[aurman]
aur =
```

nvchecker.ini

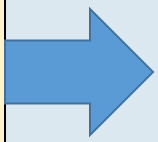


```
from lilac import *

build_prefix = 'extra-x86_64'
pre_build = aur_pre_build
post_build = aur_post_build

if __name__ == '__main__':
  single_main()
```

aurman/lilac.py



aurman-x.y-z-any.pkg.tar.xz

Automatic Packaging with nvchecker & lilac

python-svgwrite-x.y-z-any.pkg.tar.xz

Case II: pull packages from upstream & push to AUR

```
[python-svgwrite]
pypi = svgwrite
```

nvchecker.ini

```
from lilaclib import *

build_prefix = 'extra-x86_64'

def pre_build():
  pypi_pre_build(depends=['python-parsing'])

def post_build():
  pypi_post_build()
  update_aur_repo()
```

python-svgwrite/lilac.py

Maintainer: lilydjwg (lilac)

COSCUP 2018

archlinux.tw

Missing features

- Check version from multiple sources
- Dependency inference
- Support for *-testing and *-staging repositories
  - Building packages for multiple repositories
  - Handling multiple repositories
  - Package movement – staging → testing → stable
- Cyclic dependency (?)

Building for *-{testing,staging}

- Layout expansion

extra-x86_64-build

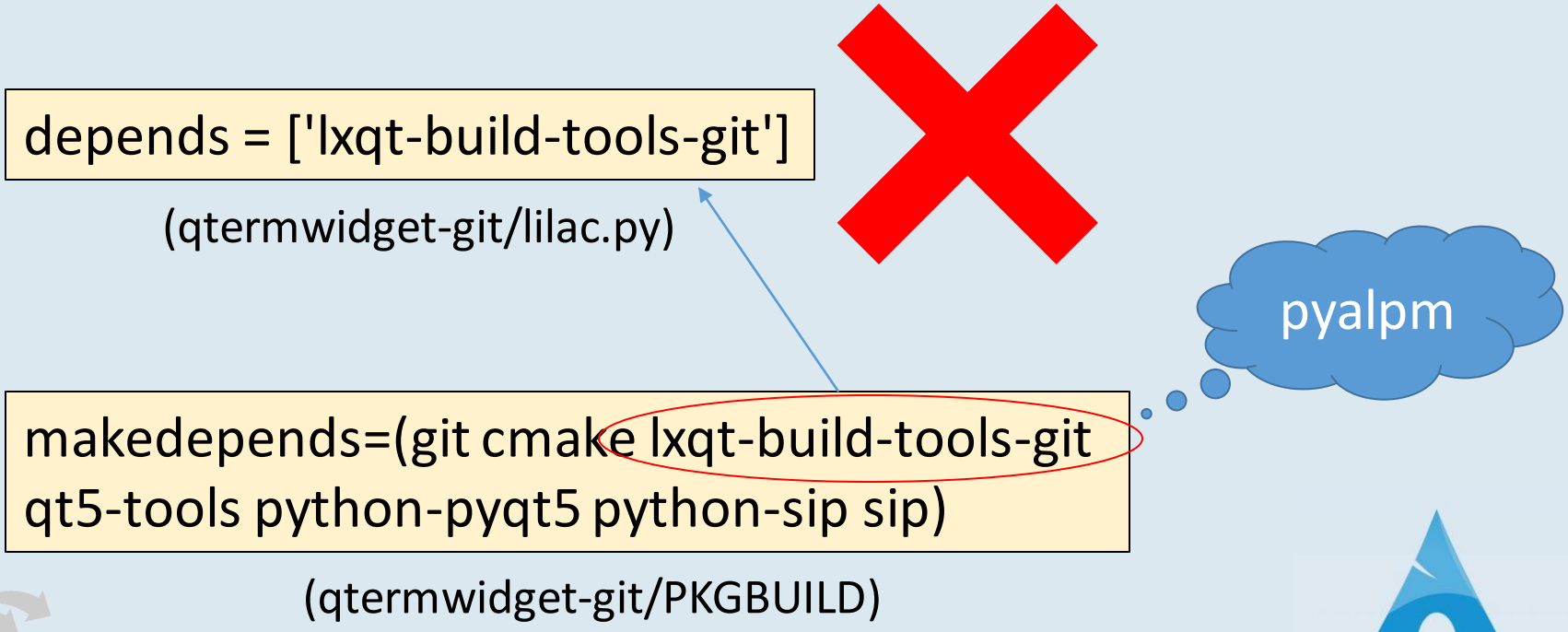
```
aurman
└── PKGBUILD
```

```
aurman
├── yan12125
│   └── PKGBUILD
└── yan12125-staging
    └── PKGBUILD
```

staging-x86_64-build

Dependency inference

- Currently lilac requires additional declarations for non-official dependencies



How can I help?

- Report bugs
- Become a tester
- Applying as a Trusted User (TU)
- Submit packages to archlinuxcn/repo
- Join Arch Linux Taiwan and materialize dreams!

Thanks!