

下一代如意包管理器进展介绍

NickCao

2023 年 6 月 27 日

假如我们需要搭建一套 RISC-V 开发环境

下一代如意包
管理器进展介
绍

NickCao

why

what

how

一、操作步骤

首先下载 `riscvQ-gnu-toolchain`

```
1 #下载riscv-gnu-toolchain
2 git clone https://github.com/riscv/riscv-gnu-toolchain
3 #更新子模块
4 git submodule update --init --recursive
```

如果下载不了，可以使用2022/3/13下载的版本：(包含gcc和quma)

链接：https://pan.baidu.com/s/1GgJ4Xo7tb_DYJB3Wjsataw

提取码：b8cr

需要大概6.65GB的磁盘

然后 `配置环境Q`（安装依赖）

```
1 | sudo apt-get install autoconf automake autotools-dev curl libmpc-dev libmpfr-dev libgmp-dev gawk build-ess
```

再编译安装 `riscv-Qgnu-toolchain`

进入到 `riscv-gnu-toolchain` 的文件目录下，执行

```
1 | cd riscv-gnu-toolchain
2 ./configure --prefix=/opt/riscv
3 #--prefix后面是你想要安装的位置，opt文件夹下默认是应用程序
4 | sudo make -jN #考虑到opt文件夹创建文件和删除文件需要用root权限
```

编译大概40min。

假如我们需要搭建一套 RISC-V 开发环境

下一代如意包
管理器进展介
绍

NickCao

why

what

how

上述编译会默认生成riscv64-unknown-linux-gnu-gcc版本的交叉编译器，但我们需要riscv64-unknown-elf-gcc。
riscv gcc可以编译成以下几个版本：

```
1 #riscv32-unknown-elf-gcc
2 #这是Newlib交叉编译器，针对riscv32架构
3 ./configure --prefix=/opt/riscv32 --with-arch=rv32imc --with-abi=xxx
4 make
5 #--mabi可以选择如下：
6 #ilp32
7 #ilp32f
8 #ilp32d
9
10 #riscv64-unknown-elf-gcc
11 #这是Newlib交叉编译器，针对riscv64架构，编译riscv-tools需要用到
12 #默认64位 因此可以不要后面的内容
13 ./configure --prefix=/opt/riscv64
14 make
15
```

最后添加环境变量（能在任何路径下使用该应用/软件）

```
1 #进入 ~/.bashrc
2 vim ~/.bashrc
3 #然后添加语句
4 export PATH=/opt/riscv/bin:$PATH
5 #保存并更新环境
6 :wq
7 source ~/.bashrc
```

为什么要用包管理

下一代如意包 管理器进展介 绍

NickCao

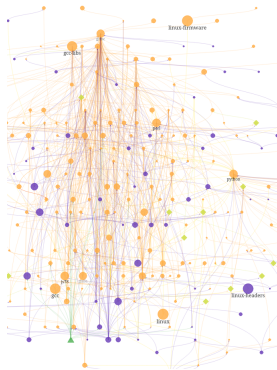
why

另一种可能

```
ryu1 package add riscv-gnu-toolchain
```

方便快捷 无需处理依赖¹或编译
生命周期 随时获得最新版本
质量控制 没有配置出错的后顾之忧

^a图为 pacvis 绘制的软件包依赖关系图



为什么又要做一个包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

Conda

- 依赖解析速度慢
- 打包质量堪忧

解决 **Anaconda3** solving environment 巨慢的方法，亲测有效！！

最近在做毕设，准备做一个基于深度学习的 **MOT** 项目，python开发，coding期间由于个人需要，所以使用Anaconda3配置了很多虚拟环境，这其中踩了不少坑，比较简单的下载慢等问题可以直接使用添加国内镜像源和下载whl文件解决，但是最让我受不了的问题就是anaconda3每次添加包的依赖或者更新包的时间实在是太久了！！！顶着solving environment的龟速做了几天实在受不了，遂去网上查找有没有解决这个问题的办法，这一查还真给我查到了！现将亲测了的解决方法中最有效的贴在下文。

1. solving environment为什么会越来越慢？

根据[原博](#)的解释以及我查阅的相关资料，这是由于conda在新安装一个包或者更新包时需要搜索当前环境中所有的包的依赖空间，以找到满足所有依赖项的版本，随着用户安装的包越来越多，这个需要搜索的依赖空间也越来越大，导致solving environment需要的时间也越来越长。

为什么又要做一个包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

Docker¹

- 无法使用主机上已经安装的工具
- 访问 GPU 等硬件往往需要复杂的配置

Spack/Gentoo Prefix

- 多不提供二进制包，仍需要编译
- 使用较为繁琐，学习曲线陡峭

¹并不是包管理

我们想要怎样的包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

简单易于使用 无需学习即可快速上手

跨发行版工作 无论是 Ubuntu、CentOS 还是 WSL 都能使用

建立隔离环境 不同项目开发环境互不干扰

可用软件包多 开发各类项目都能得心应手

多年后还工作 无论过了多久，同样的开发环境仍可重现

问题

Ubuntu 21.10 执行 `apt-get update` 后会报错：

```
1  xx Release 404 Not Found [IP: xxx]
2  E: 仓库 "xx Release" 没有 Release 文件。
3  N: 无法安全地用该源进行更新，所以默认禁用该源。
4  N: 参见 apt-secure(8) 手册以了解仓库创建和用户配置方面的细节。
```

尝试更换源（ `/etc/apt/sources.list` ）之后，还是一样的错误。

- 清华大学开源软件镜像站：<https://mirrors.tuna.tsinghua.edu.cn/help/ubuntu/>

由于使用的是 Ubuntu 21.10，在查询 [Ubuntu 生命周期和发布节奏](#) 后发现，Ubuntu 21.10 生命周期截止到 2022.07。（还有 Ubuntu 21.04，生命周期截止时间是 2022.01）

当前是 2022.10，已经超出生命结束周期（EOL）时间了。

我们有这样的包管理吗

下一代如意包
管理器进展介
绍

NickCao

why
what
how

有

Nix

- 跨发行版，乃至 macOS 和 OpenBSD 工作
- 可以建立隔离环境，同个软件包的不同版本也可共存
- 源码与二进制混合发行，便于自定义
- 目前为世界上软件包数量遥遥领先的发行版
- 十五年前的软件包依然工作²

但是

学习曲线过于陡峭，non-FHS 的设计与闭源软件等并不兼容

²<https://blinry.org/nix-time-travel/>

我们如何构建一个这样的包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

Mom, can we have **PACKAGE MANAGER**?



No. There is **PACKAGE MANAGER** At Home



PACKAGE MANAGER At home...



我们如何构建一个这样的包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

Git

- 包管理的本质就是把指定文件放到指定位置
- 使用环境: `git checkout base`
- 创建环境: `git checkout -b develop`
- 安装软件: `git cherry-pick gcc`
- 保存环境: `git commit -m "installed gcc"`
- 分享环境: `git pull/push`

但是

Git 并不适合管理二进制文件

我们如何构建一个这样的包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

OSTree²

- 专为二进制文件设计的 Git
- content addresses 存储，自动去重
- 可以使用容器镜像服务存储数据

但是

里面的东西哪里来

元包管理

不如我们把现有的东西塞进去

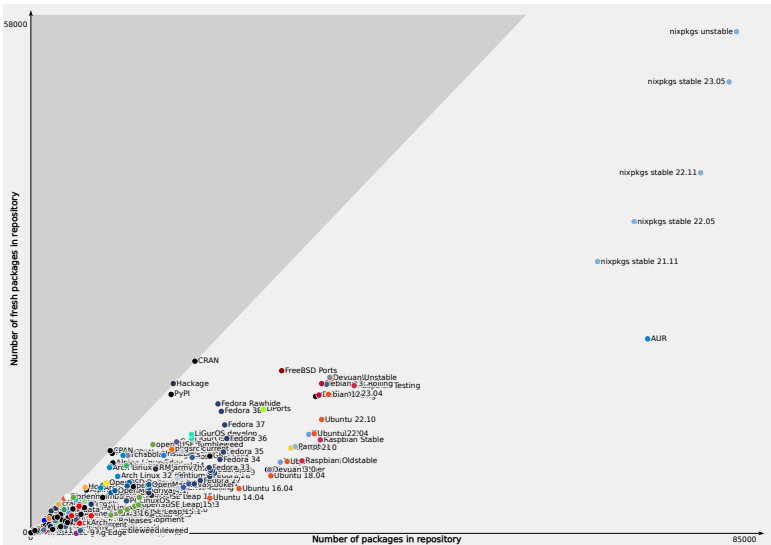
²<https://ostreedev.github.io/ostree/>

现有的东西

下一代如意包 管理器进展介 绍

NickCao

how



我们试图构建一个这样的包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

第一次尝试

直接把一个现有的发行版解压到一个目录里

```
$ tar xf archlinux.tar.gz -C /opt/arch/  
$ /opt/arch/bin/ls  
bash: ls: cannot execute: required file not found  
# 动态链接的二进制程序需要解释器才能执行  
$ patchelf --print-interpreter /opt/arch/bin/ls  
/lib64/ld-linux-x86-64.so.2
```

我们好像构建出了一个这样的包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

第二次尝试

解压后再做点修改

```
$ patchelf --set-interpreter \  
    /opt/arch/lib64/ld-linux-x86-64.so.2 \  
    /opt/arch/bin/ls  
$ /opt/arch/bin/ls  
/opt/arch/bin/ls: error while loading shared  
libraries: libcap.so.2: cannot open shared object  
file: No such file or directory  
$ patchelf --set-rpath /opt/arch/lib /opt/arch/bin/ls  
$ /opt/arch/bin/ls  
Documents Downloads Pictures Projects
```

我们好像构建出了一个这样的包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

但是

软件包中不只有 ELF，还有头文件、资源文件、配置文件...

chroot

让文件在它原本该在的地方

- 首先把一个发行版的镜像解压到某个地方
- 然后 chroot 进去
- 我们成功发明了：Docker

为什么我们不喜欢 Docker

我们想要隔离的环境，但我们不想要隔离

我们如何构建一个这样的包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

我们需要一个支持在 prefix 下运行的发行版

Gentoo Prefix

To bring out the virtues of Gentoo Linux on different operating systems, the Gentoo Prefix project develops and maintains a way of installing Gentoo systems in a non-standard location, designated by a "prefix".

Fedora (or generally, rpm)

[https://docs.fedoraproject.org/en-US/
packaging-guidelines/#_relocatable_packages](https://docs.fedoraproject.org/en-US/packaging-guidelines/#_relocatable_packages)

我们如何构建一个这样的包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

我们也可以先将就一下

bubblewrap

Low-level unprivileged sandboxing tool used by Flatpak and similar projects

```
$ bwrap \  
  --bind /opt/arch / \  
  --bind /home /home \  
  /usr/bin/ls  
Documents Downloads Pictures Projects
```

我们如何构建一个这样的包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

- 简单易于使用
- 跨发行版工作 ✓
- 建立隔离环境 ✓
- 可用软件包多 ✓
- 多年后还工作

我们如何构建一个这样的包管理

下一代如意包
管理器进展介
绍

NickCao

why

what

how

简单易于使用

- 为各发行版建立一套统一的基线环境
- 为各种语言和框架提供预构建的开发套件

多年后还工作

- 为各发行版提供历史镜像
- 使用类似 Dockerfile 的形式创建开发环境

我们如何构建一个这样的包管理

下一代如意包
管理器进展介
绍

NickCao

why
what
how

- 保存每个发行版的软件仓库每天的状态
- 基于发行版仓库元数据锁定文件哈希
- 对外提供和普通镜像源一致的 http 访问
- 无需修改包管理器，仅修改镜像地址即可回到任意一天

FROM ubuntu:2023-06-27

LANGUAGE rust haskell c++

FRAMEWORK boost qt6

PACKAGE msgpack-cxx doxygen

下一代如意包管理器

下一代如意包
管理器进展介
绍

NickCao

why

what

how

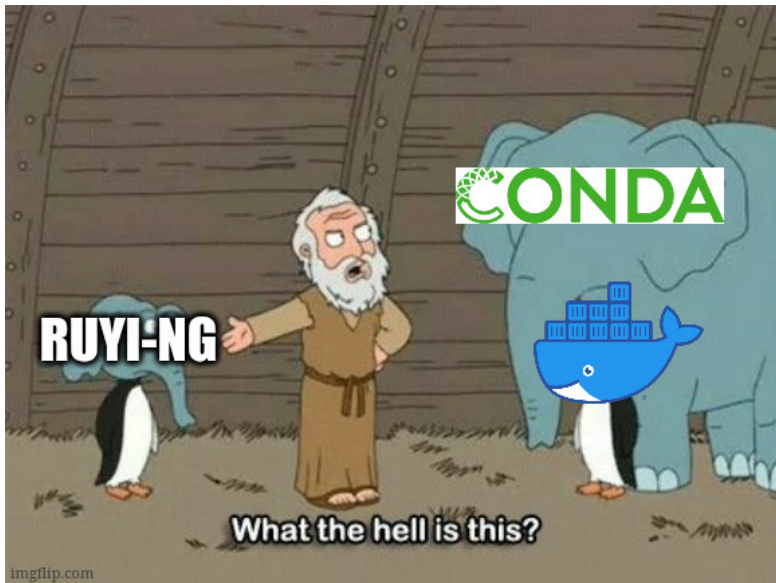
```
# 初始化如意
$ ruyi init
# 拉取基础镜像
$ ruyi pull example.com/archlinux
# 建立工作副本
$ ruyi checkout example.com/archlinux develop
# 激活开发环境
$ ruyi activate develop
# 对环境进行修改
$ ruyi framework add qt6
$ pacman -S nano
# 提交改动
$ ruyi commit develop example.com/archlinux-new
$ ruyi push example.com/archlinux-new
```

下一代如意包管理器

下一代如意包
管理器进展介
绍

NickCao

why
what
how



下一代如意包管理器

下一代如意包
管理器进展介
绍

NickCao

why











what

how

- 基于 OSTree 与 bubblewrap 打造
- 提供一个类似 Docker 与 Conda 的易用的 cli 界面
- 手动创建环境与 Ruyifile 声明式构建两种工作模式
- 长期可用的历史镜像确保可复现性

Bonus

为如意而建立的镜像存储服务与 cli 也可用于管理非如意创建的系统镜像，为目前的工作流提供帮助

REPOSITORY NAME	LAST MODIFIED	ACTIVITY ↓
 library / ubuntu-cloud	Last Thursday at 6:31 PM	
 library / archlinux	Last Thursday at 6:08 PM	
 library / revyos	06/19/2023	
 library / taoky-debian-riscv	06/14/2023	
 library / openeuler-sig-riscv	06/14/2023	

下一代如意包管理器

下一代如意包
管理器进展介
绍

NickCao

why

what

how

- 简单易于使用 ✓
- 跨发行版工作 ✓
- 建立隔离环境 ✓
- 可用软件包多 ✓
- 多年后还工作 ✓

ruyi-ng

<https://github.com/NickCao/ruyi-ng>