openEuler

openEuler可重复构建方案与机制

目录：

1. **可重复构建业界洞察**
2. **可重复构建openEuler实践**
3. **相关工具原理**
4. **进展&看板演示**
5. **下一步计划与社区运作**

可重复构建洞察

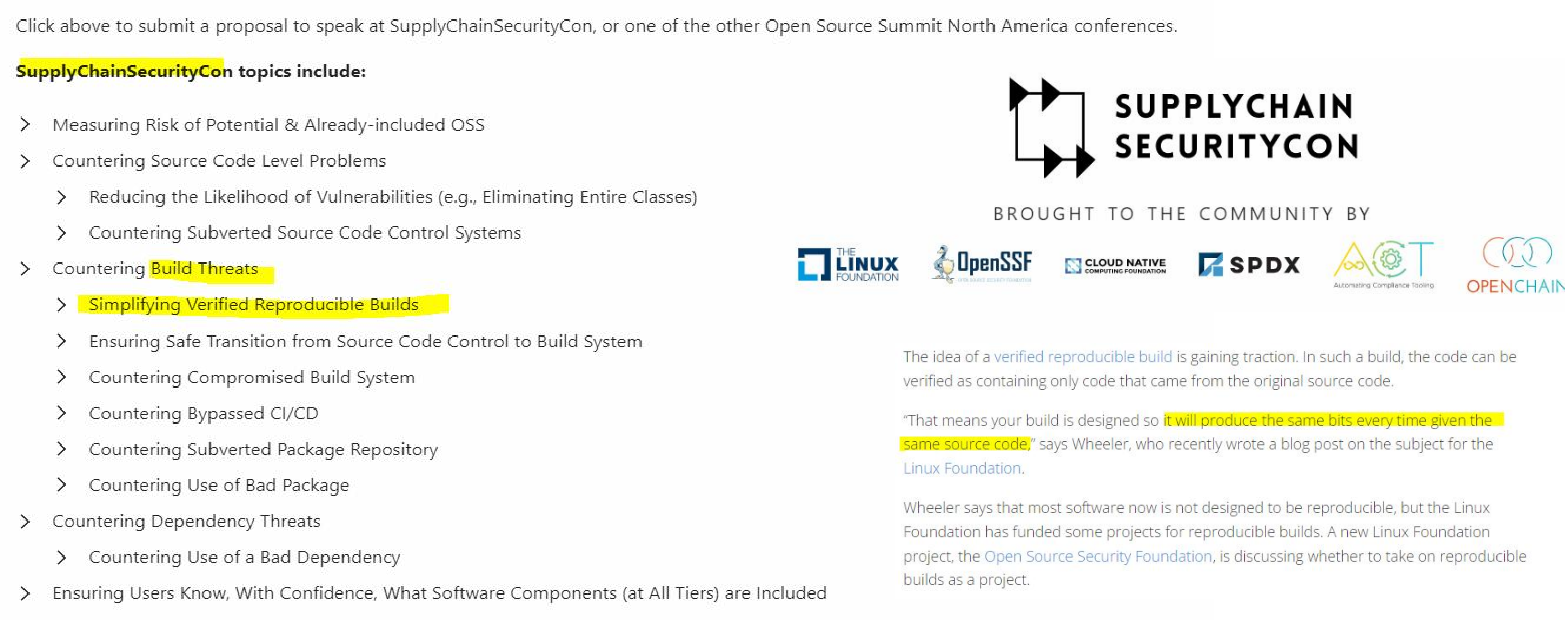
软件供应链安全提到前所未有高度、如何证明?

OpenEuler

可重复构建（ Reproducible Builds）是证明软件供应链安全的必要手段

当前已纳入supplychainsecuritycon的topics

https://events.linuxfoundation.org/open-source-summit-north-america/about/supplychainsecuritycon/



Click above to submit a proposal to speak at SupplyChainSecurityCon, or one of the other Open Source Summit North America conferences.

SupplyChainSecurityCon topics include:

- Measuring Risk of Potential & Already-included OSS
- Countering Source Code Level Problems
  - Reducing the Likelihood of Vulnerabilities (e.g., Eliminating Entire Classes)
  - Countering Subverted Source Code Control Systems
- Countering Build Threats
  - Simplifying Verified Reproducible Builds
  - Ensuring Safe Transition from Source Code Control to Build System
  - Countering Compromised Build System
  - Countering Bypassed CI/CD
  - Countering Subverted Package Repository
  - Countering Use of Bad Package
- Countering Dependency Threats
  - Countering Use of a Bad Dependency
- Ensuring Users Know, With Confidence, What Software Components (at All Tiers) are Included

SUPPLYCHAIN
SECURITYCON
BROUGHT TO THE COMMUNITY BY
THE LINUX FOUNDATION
OpenSSF
CLOUD NATIVE COMPUTING FOUNDATION
SPDX
ACT Automating Compliance Tooling
OPENCHAIN

The idea of a verified reproducible build is gaining traction. In such a build, the code can be verified as containing only code that came from the original source code.

"That means your build is designed so it will produce the same bits every time given the same source code," says Wheeler, who recently wrote a blog post on the subject for the Linux Foundation.

Wheeler says that most software now is not designed to be reproducible, but the Linux Foundation has funded some projects for reproducible builds. A new Linux Foundation project, the Open Source Security Foundation, is discussing whether to take on reproducible builds as a project.

可重复构建概念和研究

- 概念定义

对于可重复的构建，给定相同的源代码、构建环境和构建指令，任何人均可重建出比特级完全相同的指定制品。

- 现实意义

创建从代码到制品的可独立验证路径，结合已有的代码发布签名、软件仓库签名、安全启动等技术，使开源代码从生产到使用的全过程可追溯成为可能。

学术论文

- *Trusting Trust - Reflections on Trusting Trust* (1984) — Ken Thompson. (PDF)
- *Fully Countering Trusting Trust through Diverse Double-Compiling* (2005/2009) — David A. Wheeler (PDF, …)
- *Functional Package Management with Guix* (2013) — Ludovic Courtès. […]
- *Reproducible and User-Controlled Software Environments in HPC with Guix* (2015) — Ludovic Courtès, Ricardo Wurmus […]
- *in-toto: Providing farm-to-table guarantees for bits and bytes* (2019) — Santiago Torres-Arias, New York University; Hammad Afzali, New Jersey Institute of Technology; Trishank Karthik Kuppusamy, Datadog; Reza Curtmola, New Jersey Institute of Technology; Justin Cappos, New York University. (PDF)
- *Backstabber's Knife Collection: A Review of Open Source Software Supply Chain Attacks* (2020) — Marc Ohm, Henrik Plate, Arnold Sykosch, Michael Meier. (PDF)
- *Automated Localization for Unreproducible Builds* (2018) — Zhilei Ren, He Jiang, Jifeng Xuan, Zijiang Yang. (PDF)
- *Reproducible Containers* (2020) — Navarro Leija, Omar S. and Shiptoski, Kelly and Scott, Ryan G. and Wang, Baojun and Renner, Nicholas and Newton, Ryan R. and Devietti, Joseph. (…)
- *Towards detection of software supply chain attacks by forensic artifacts* — Marc Ohm, Arnold Sykosch, Michael Meier. (Link)

可重复构建带来了什么收益

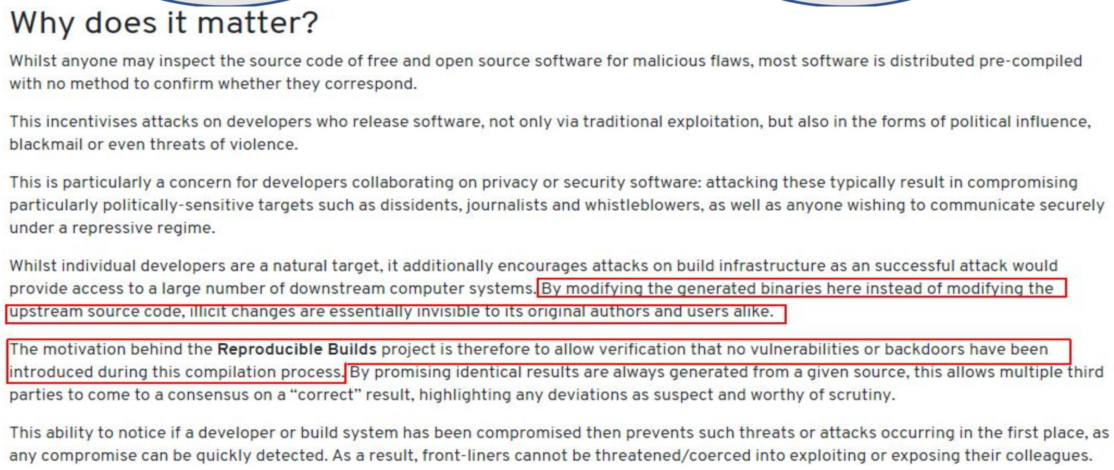
- 验证二进制未被植入后门，避免潜在安全风险；二进制质量保障
- 构建环境&构建工程能够被还原，依赖变化范围最小化、测试最小化、方便问题定位提高开发效率

可追溯性：任何人在获得授权的前提下，能够找到软件的任何变更历史和信息、不限于构建环境、构建工程、源代码、依赖信息等

Traceability
可追溯性

Reproducible
可重复性

可重复性：任何人在获得授权的前提下，能够重现从过去到现在之间任意时间点的状态



Why does it matter?

Whilst anyone may inspect the source code of free and open source software for malicious flaws, most software is distributed pre-compiled with no method to confirm whether they correspond.

This incentivises attacks on developers who release software, not only via traditional exploitation, but also in the forms of political influence, blackmail or even threats of violence.

This is particularly a concern for developers collaborating on privacy or security software: attacking these typically result in compromising particularly politically-sensitive targets such as dissidents, journalists and whistleblowers, as well as anyone wishing to communicate securely under a repressive regime.

Whilst individual developers are a natural target, it additionally encourages attacks on build infrastructure as an successful attack would provide access to a large number of downstream computer systems. By modifying the generated binaries here instead of modifying the upstream source code, illicit changes are essentially invisible to its original authors and users alike.

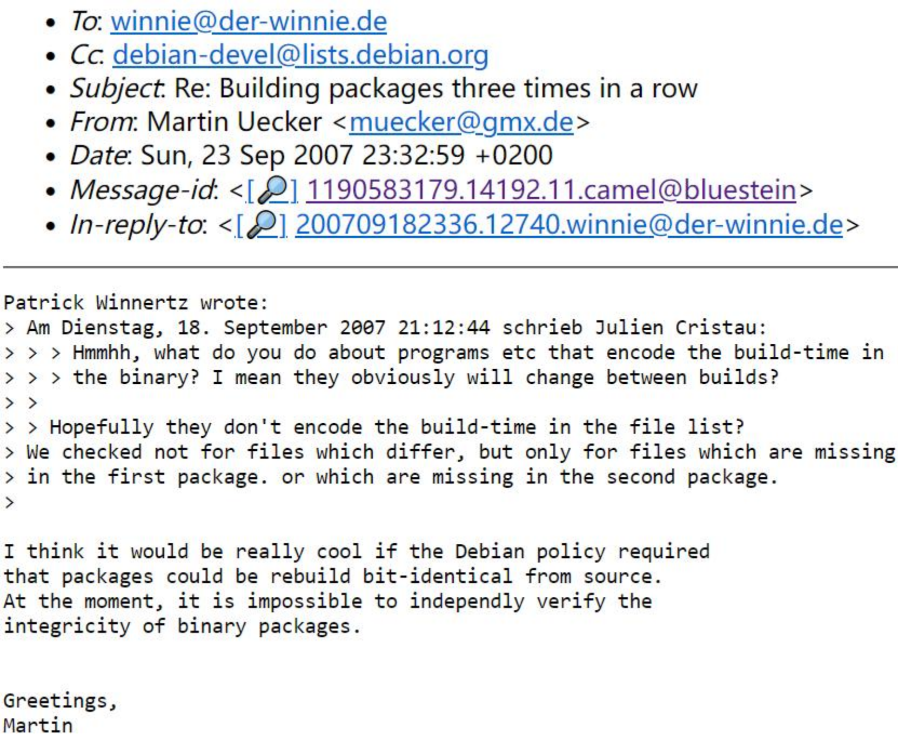
The motivation behind the **Reproducible Builds** project is therefore to allow verification that no vulnerabilities or backdoors have been introduced during this compilation process. By promising identical results are always generated from a given source, this allows multiple third parties to come to a consensus on a "correct" result, highlighting any deviations as suspect and worthy of scrutiny.

This ability to notice if a developer or build system has been compromised then prevents such threats or attacks occurring in the first place, as any compromise can be quickly detected. As a result, front-liners cannot be threatened/coerced into exploiting or exposing their colleagues.

可重复构建社区的历史

可重复构建并非新生事物：

- 1992 年 GNU 工具链已有工具对此提供初步支持
- 2000 年 Debian 开发者提出了关于内嵌时间戳影响可重复性的技术讨论
- 2007 年 Debian 项目开始了全面支持可重复构建的思考
- 2013 年 Snowden 事件使社会对这个问题的关注度大幅提升；研究人员通过修改的 Xcode 实现了攻击，随后 XcodeGhost 被发现。
- 同年的Debian 开发者会议(Debconf13)过程中，可重复构建专题启动到八月完成首轮盲测5240个软件，可重复率仅24%



- *To*: winnie@der-winnie.de
- *Cc*: debian-devel@lists.debian.org
- *Subject*: Re: Building packages three times in a row
- *From*: Martin Uecker <muecker@gmx.de>
- *Date*: Sun, 23 Sep 2007 23:32:59 +0200
- *Message-id*: <[🔎] 1190583179.14192.11.camel@bluestein>
- *In-reply-to*: <[🔎] 200709182336.12740.winnie@der-winnie.de>

```
Patrick Winnertz wrote:
> Am Dienstag, 18. September 2007 21:12:44 schrieb Julien Cristau:
> > > Hmmhh, what do you do about programs etc that encode the build-time in
> > > the binary? I mean they obviously will change between builds?
> >
> > Hopefully they don't encode the build-time in the file list?
> We checked not for files which differ, but only for files which are missing
> in the first package. or which are missing in the second package.
>

I think it would be really cool if the Debian policy required
that packages could be rebuild bit-identical from source.
At the moment, it is impossible to independly verify the
integricity of binary packages.


Greetings,
Martin
```

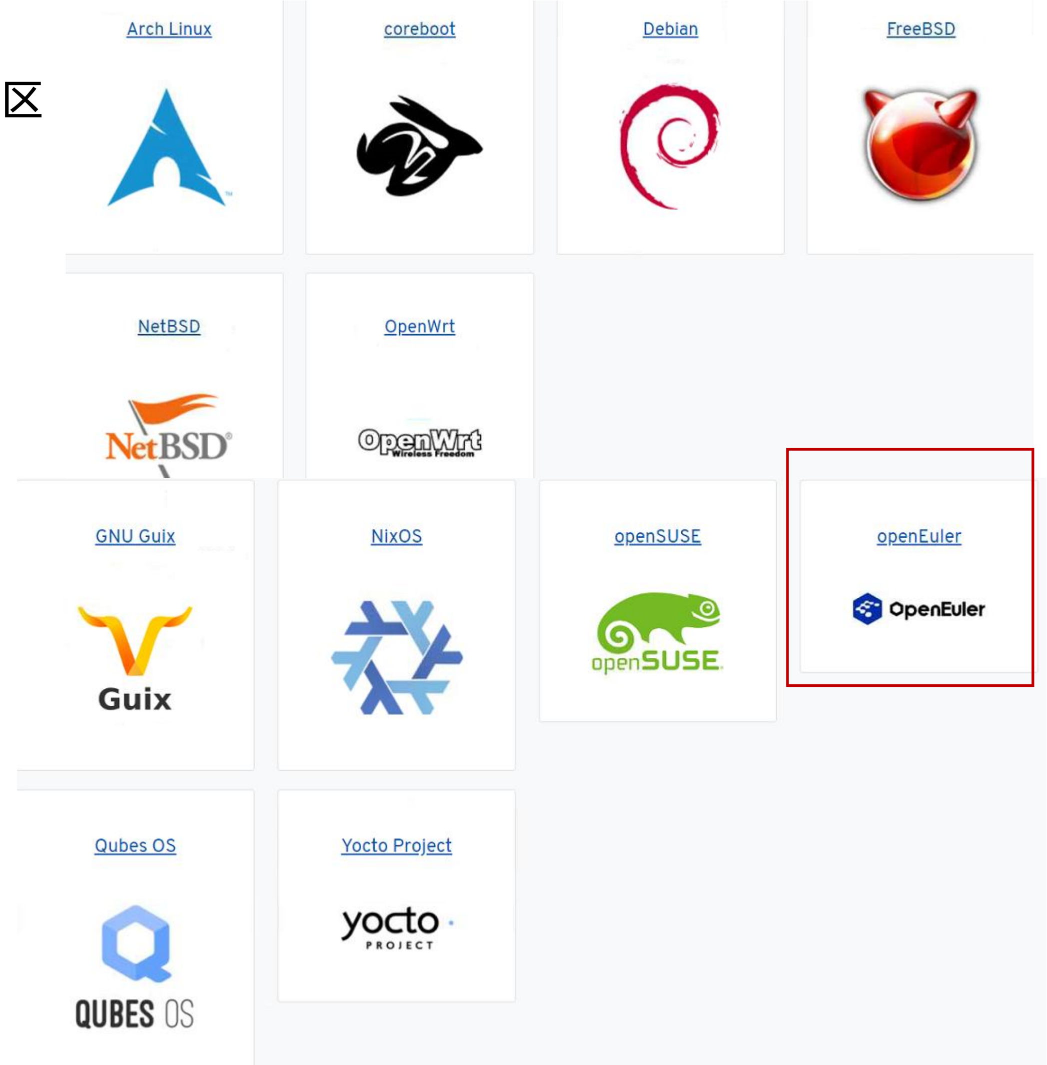
进一步推动：

- 2014年 binutils 代码确定性生成补丁合入，二轮盲测6887个软件，达成率提高至 67%；这一年的Debian开发者会议(Debconf14)上敲定了可重复构建元数据格式，完成了基础集成工具的功能打样
- 2015年 FOSDEM15和Debconf15上，相关工具支持纳入 Debian 官方基础设施，并作为新版 Release Goal，此时全仓库达成率已推进至 83%
- 这一年还首次实现了 Firefox 的可重复构建；Coreboot, OpenWrt, NetBSD, FreeBSD, Archlinux 相继加入计划
- 2016-2017年，可重复构建纳入Debian社区官方打包规范 (4.1.0版本)，同时全仓库达成率推进至90%
- 2022年发布的Debian 11中，全仓库达成率94.8%(amd64, 31500 packages)

谁参与了可重复构建社区建设

https://reproducible-builds.org/citests/

主要支持社区


Arch Linux

coreboot

Debian

FreeBSD

NetBSD

OpenWrt

GNU Guix

NixOS

openSUSE

openEuler

Qubes OS

Yocto Project

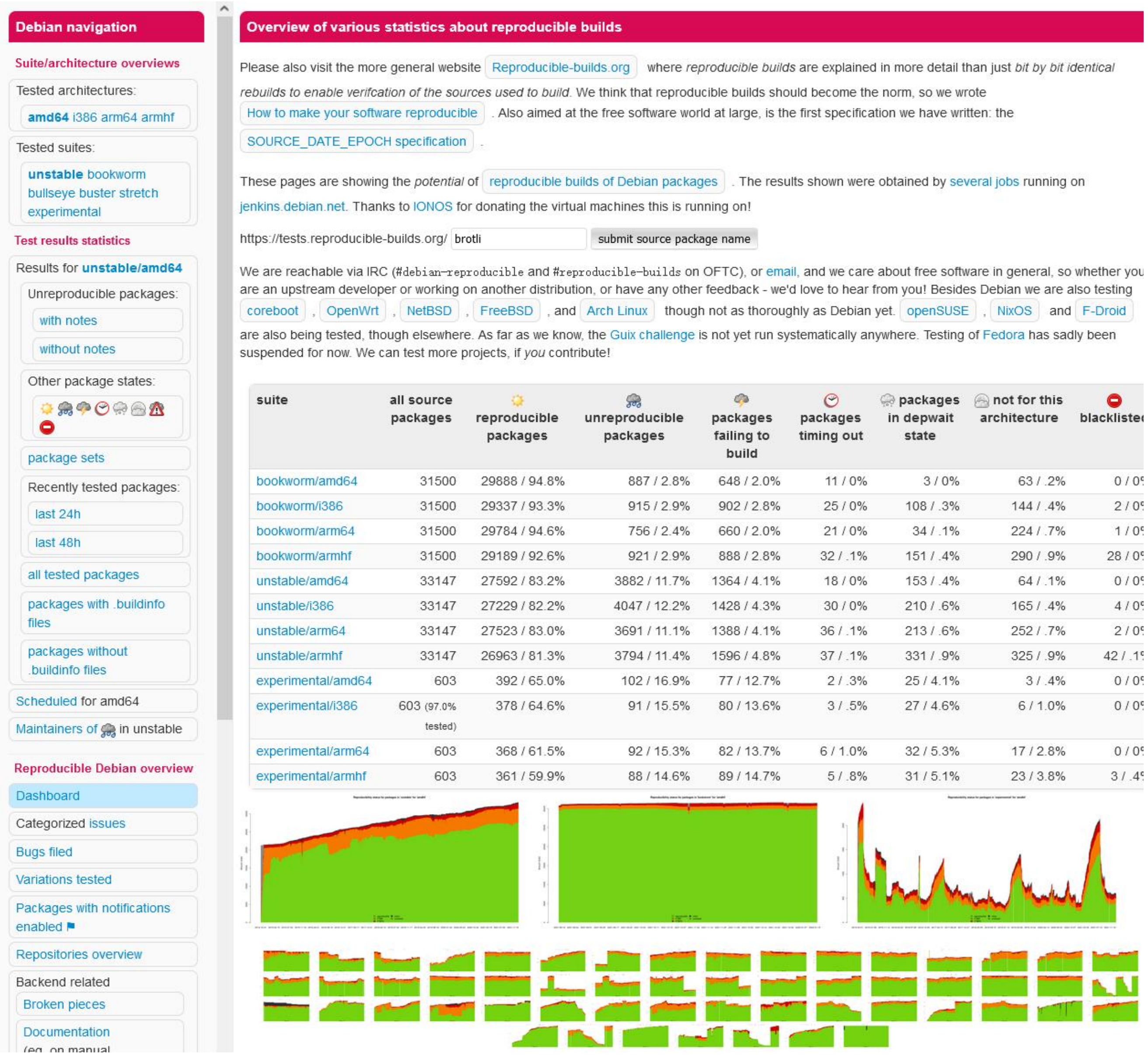
合作社区

Reproducible builds are a set of software development practices that create an independently-verifiable path from source to binary code. (more)

项目	可重构建比例	总包数量
Debian	94.8%	30192
coreboot	100%	167
NetBSD	96.6%	60
openSUSE	95.9%	13929
NixOS	99.87%	1572 core package
Arch Linux	80.7%	10703
Fedora	93%	845 2017年RPM系列OS 由于投资问题、项目停滞)
openEuler	98%	核心1800+仓

社区现有可重复构建工具集和CI看板

名称	功能
diffoscope	差异比较工具
trydiffoscope	使用diffoscope的在线服务平台
disorderfs	非确定性文件系统注入工具
strip-nondeterminism	非确定性编译系统后处理工具
reprotest	可重复构建验证工具
rebuilderd	发行版仓库监控和验证工具
archlinux-repro	Archlinux验证后端
reproducible-build-maven-plugin	Apache Maven 构建系统插件
sbt-reproducible-builds	Scala sbt 构建系统插件



Debian navigation

Suite/architecture overviews

Tested architectures:
amd64 i386 arm64 armhf

Tested suites:
unstable bookworm bullseye buster stretch experimental

Test results statistics

Results for **unstable/amd64**

Unreproducible packages:
- with notes
- without notes

Other package states:
☀ 🌧 ⛈ ⏰ 🌨 ⛅ ⚠ ⛔

package sets

Recently tested packages:
- last 24h
- last 48h

all tested packages

packages with .buildinfo files

packages without .buildinfo files

Scheduled for amd64

Maintainers of 🌧 in unstable

Reproducible Debian overview

Dashboard

Categorized issues

Bugs filed

Variations tested

Packages with notifications enabled ⚑

Repositories overview

Backend related
- Broken pieces
- Documentation (eg. on manual

Overview of various statistics about reproducible builds

Please also visit the more general website Reproducible-builds.org where *reproducible builds* are explained in more detail than just *bit by bit identical rebuilds to enable verifcation of the sources used to build*. We think that reproducible builds should become the norm, so we wrote How to make your software reproducible . Also aimed at the free software world at large, is the first specification we have written: the SOURCE_DATE_EPOCH specification .

These pages are showing the *potential* of reproducible builds of Debian packages . The results shown were obtained by several jobs running on jenkins.debian.net. Thanks to IONOS for donating the virtual machines this is running on!

https://tests.reproducible-builds.org/ brotli submit source package name

We are reachable via IRC (`#debian-reproducible` and `#reproducible-builds` on OFTC), or email, and we care about free software in general, so whether you are an upstream developer or working on another distribution, or have any other feedback - we'd love to hear from you! Besides Debian we are also testing coreboot , OpenWrt , NetBSD , FreeBSD , and Arch Linux though not as thoroughly as Debian yet. openSUSE , NixOS and F-Droid are also being tested, though elsewhere. As far as we know, the Guix challenge is not yet run systematically anywhere. Testing of Fedora has sadly been suspended for now. We can test more projects, if *you* contribute!

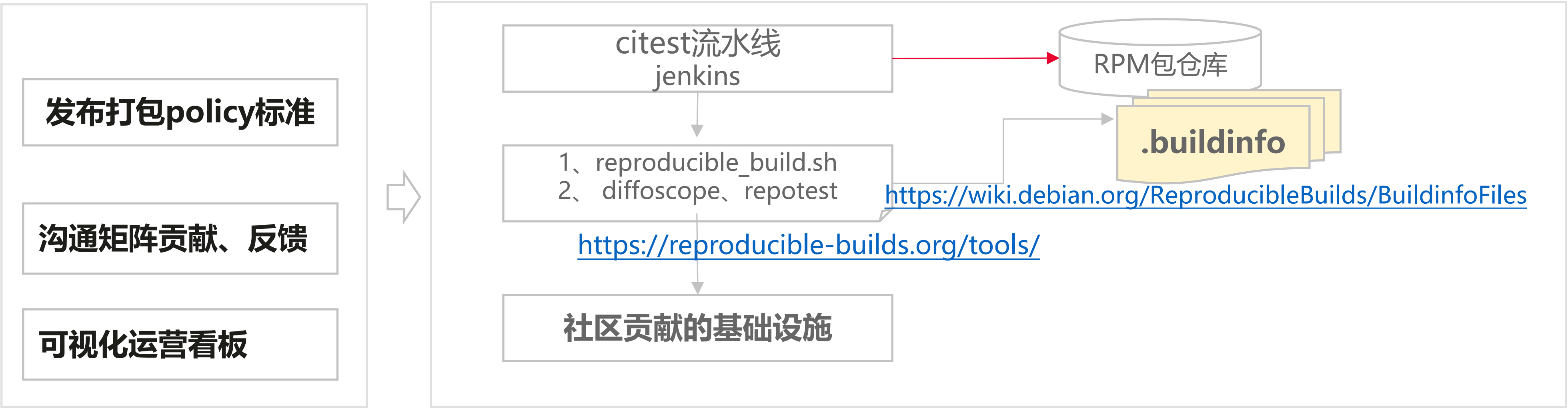
suite	all source packages	reproducible packages	unreproducible packages	packages failing to build	packages timing out	packages in depwait state	not for this architecture	blackliste
bookworm/amd64	31500	29888 / 94.8%	887 / 2.8%	648 / 2.0%	11 / 0%	3 / 0%	63 / .2%	0 / 0
bookworm/i386	31500	29337 / 93.3%	915 / 2.9%	902 / 2.8%	25 / 0%	108 / .3%	144 / .4%	2 / 0
bookworm/arm64	31500	29784 / 94.6%	756 / 2.4%	660 / 2.0%	21 / 0%	34 / .1%	224 / .7%	1 / 0
bookworm/armhf	31500	29189 / 92.6%	921 / 2.9%	888 / 2.8%	32 / .1%	151 / .4%	290 / .9%	28 / 0
unstable/amd64	33147	27592 / 83.2%	3882 / 11.7%	1364 / 4.1%	18 / 0%	153 / .4%	64 / .1%	0 / 0
unstable/i386	33147	27229 / 82.2%	4047 / 12.2%	1428 / 4.3%	30 / 0%	210 / .6%	165 / .4%	4 / 0
unstable/arm64	33147	27523 / 83.0%	3691 / 11.1%	1388 / 4.1%	36 / .1%	213 / .6%	252 / .7%	2 / 0
unstable/armhf	33147	26963 / 81.3%	3794 / 11.4%	1596 / 4.8%	37 / .1%	331 / .9%	325 / .9%	42 / .1
experimental/amd64	603	392 / 65.0%	102 / 16.9%	77 / 12.7%	2 / .3%	25 / 4.1%	3 / .4%	0 / 0
experimental/i386	603 (97.0% tested)	378 / 64.6%	91 / 15.5%	80 / 13.6%	3 / .5%	27 / 4.6%	6 / 1.0%	0 / 0
experimental/arm64	603	368 / 61.5%	92 / 15.3%	82 / 13.7%	6 / 1.0%	32 / 5.3%	17 / 2.8%	0 / 0
experimental/armhf	603	361 / 59.9%	88 / 14.6%	89 / 14.7%	5 / .8%	31 / 5.1%	23 / 3.8%	3 / .4

看业界Debian可重复构建方案

1、有一个严格发布包的policy 比如time-stamps，debian/rules， 打包格式，copyright,changelog格式，依赖的严格定义方式

2、有成熟的运作体系：有组织、有沟通矩阵、邮件反馈、推动社区改进等Join the Reproducible Builds group ， #reproducible-changes

3、有成体系的工具链：比较工具、自动生成采集信息buildInfo基于buildInfo重复构建、基础设施、可重复构建状态看板



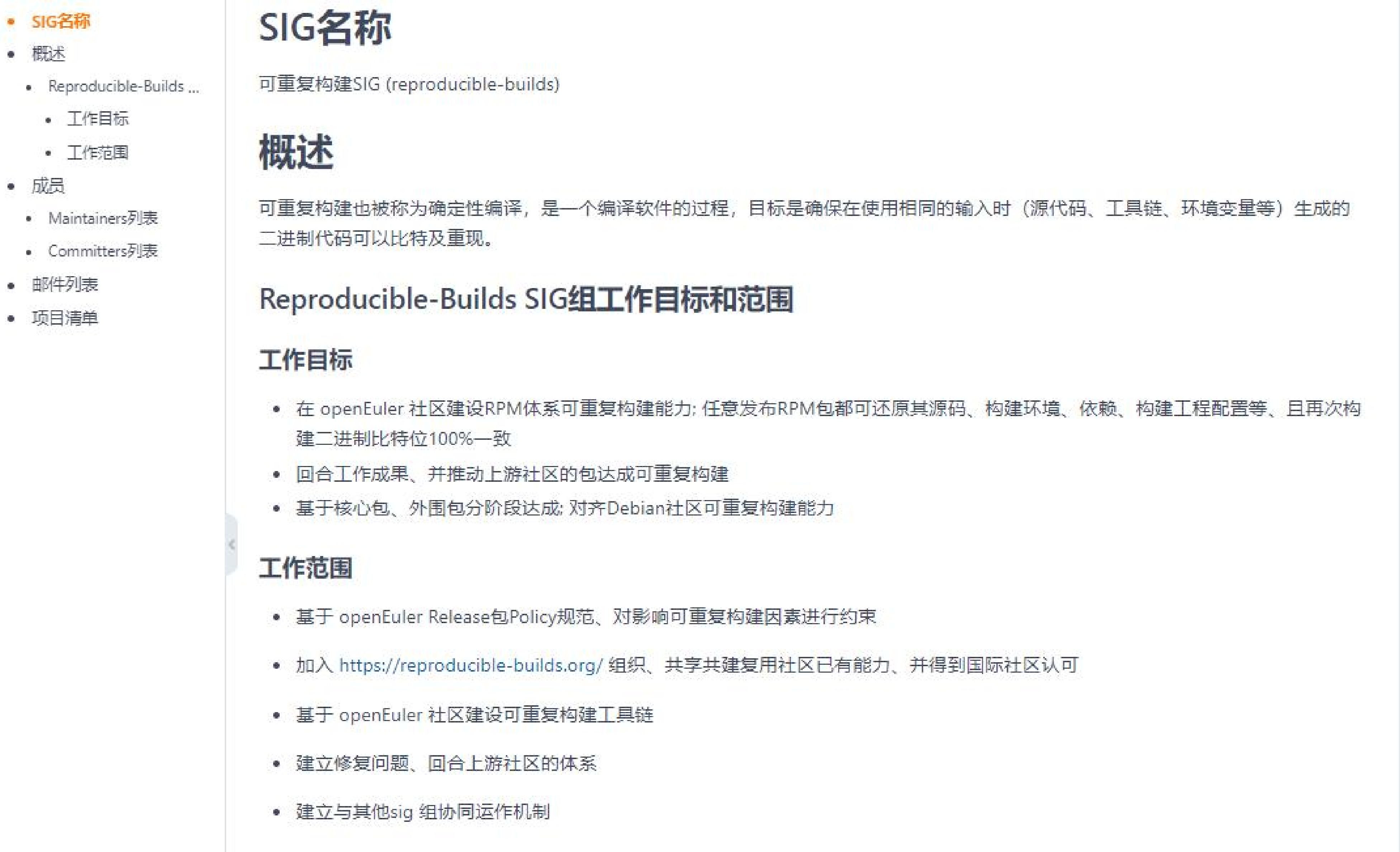
优点：Debian的可重复整体方案并不是只是给debian用的，可以复制到其它操作系统； 建设比较早、社区认可

缺点：工具链基于jenkins1.0+ JJB 插件+ shell脚本+工具的方式；相对比较古老

openEuler

openEuler可重复构建实践

openEuler可重复构建SIG运作：工作目标、范围



- SIG名称
- 概述
  - Reproducible-Builds ...
    - 工作目标
    - 工作范围
- 成员
  - Maintainers列表
  - Committers列表
- 邮件列表
- 项目清单

SIG名称

可重复构建SIG (reproducible-builds)

概述

可重复构建也被称为确定性编译，是一个编译软件的过程，目标是确保在使用相同的输入时（源代码、工具链、环境变量等）生成的二进制代码可以比特及重现。

Reproducible-Builds SIG组工作目标和范围

工作目标

- 在 openEuler 社区建设RPM体系可重复构建能力; 任意发布RPM包都可还原其源码、构建环境、依赖、构建工程配置等、且再次构建二进制比特位100%一致
- 回合工作成果、并推动上游社区的包达成可重复构建
- 基于核心包、外围包分阶段达成; 对齐Debian社区可重复构建能力

工作范围

- 基于 openEuler Release包Policy规范、对影响可重复构建因素进行约束
- 加入 https://reproducible-builds.org/ 组织、共享共建复用社区已有能力、并得到国际社区认可
- 基于 openEuler 社区建设可重复构建工具链
- 建立修复问题、回合上游社区的体系
- 建立与其他sig 组协同运作机制

https://gitee.com/openeuler/community/tree/master/sig/sig-reproducible-builds

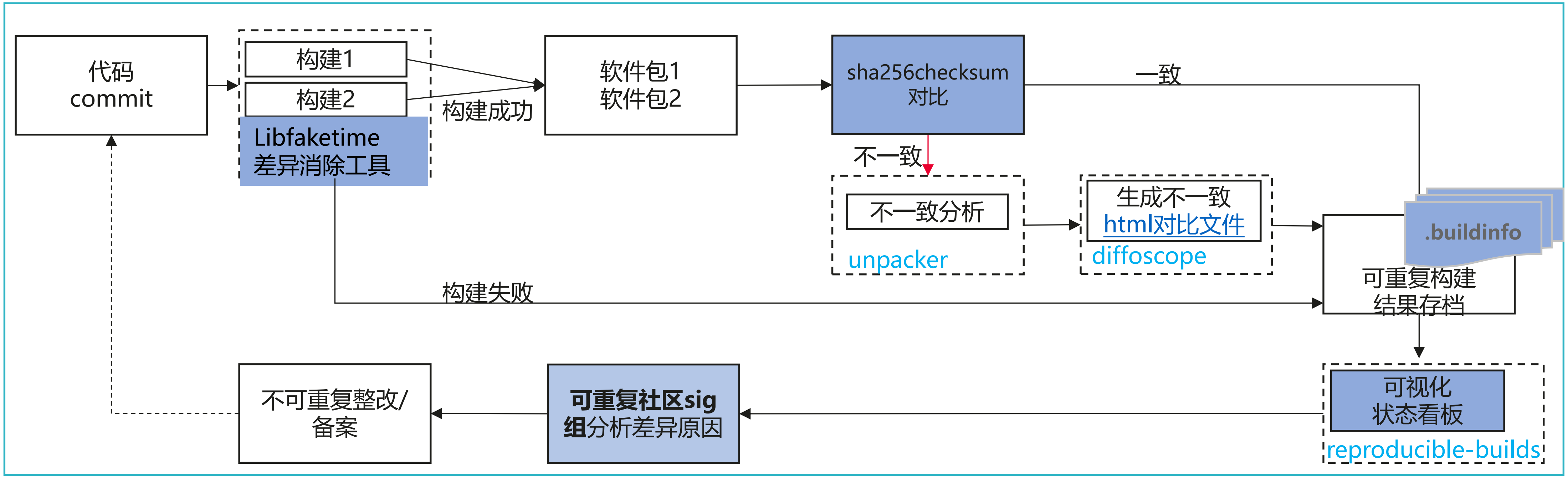
openEuler可重复构建整体方案

OpenEuler

措施

- **组织运作：**建立推动社区贡献组织：可重复问题可视、提交issue、修复、 回合验证体系
- **工具链建设：**可重复构建调度；差异消除、二进制对比等开源工具引入； buildInfo自动采集；状态看板建设

实践应用



代码
commit

构建1

构建2

Libfaketime
差异消除工具

构建成功

构建失败

软件包1
软件包2

sha256checksum
对比

一致

不一致

不一致分析

unpacker

生成不一致
html对比文件

diffoscope

.buildinfo

可重复构建
结果存档

可视化
状态看板

reproducible-builds

可重复社区sig
组分析差异原因

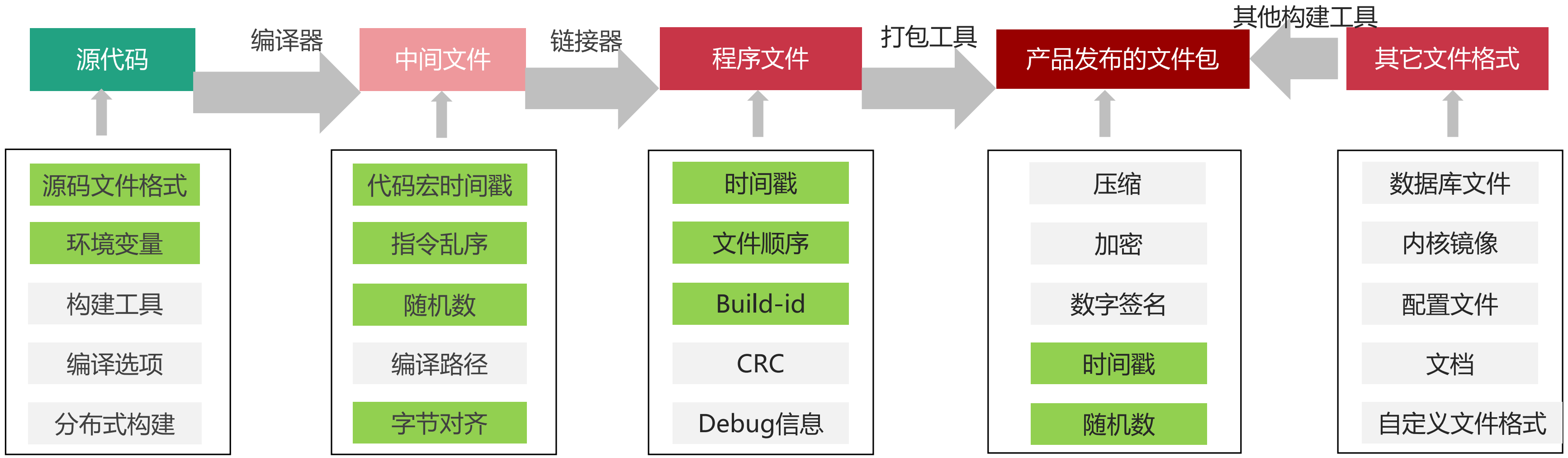
不可重复整改/
备案

应用效果

https://reproducible-builds.openeuler.org/
https://gitee.com/openeuler/reproducible-builds
https://reproducible-builds.org/citests/

为什么可重复构建会有差异

- 在代码构建期间，从源代码码到产品发布的二进制包，中间每一个步骤，每一个构建工具都有可能引入二进制差异，而这些差异经过逐步放大，导致最终发布的二进制包每次编译都不相同，而且差异非常巨大。
- 导致构建差异的因素有很多，包含时间戳，随机数，文件乱序，这些差异都是在构建过程中生成的，数字签名可以证明源码和二进制的唯一性，但是无法证明源码与二进制之前的一致性，或者证明源码与二进制一致性的工作量非常大、技术难度高。

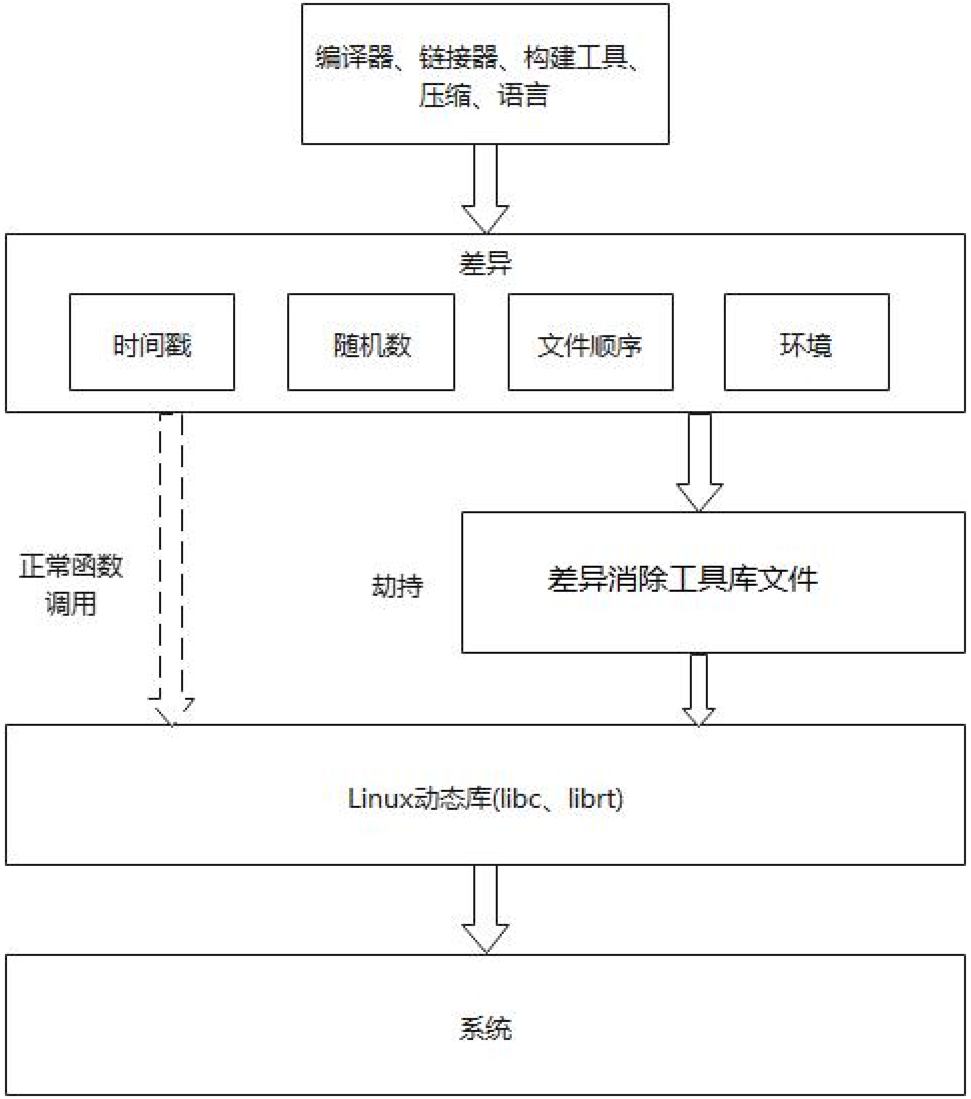


源代码 → 编译器 → 中间文件 → 链接器 → 程序文件 → 打包工具 → 产品发布的文件包 ← 其他构建工具 ← 其它文件格式

源代码	中间文件	程序文件	产品发布的文件包	其它文件格式
源码文件格式	代码宏时间戳	时间戳	压缩	数据库文件
环境变量	指令乱序	文件顺序	加密	内核镜像
构建工具	随机数	Build-id	数字签名	配置文件
编译选项	编译路径	CRC	时间戳	文档
分布式构建	字节对齐	Debug信息	随机数	自定义文件格式

- 差异消除能力：支持主流操作系统上时间戳、随机数、文件排序等差异消除，差异消除覆盖比例超过80%;
- 支持多种编程语言:已支持C\C++, Java, C#和Python等；支持随机差异消除:已支持 readdir, 随机数, inode, DevId等
- 解包和差异分析工具来分析通用的压缩、打包格式文件，并针对差异文件进行差异识别与差异分析。

基于开源Libfaketime差异消除工具原理



机制

- 差异消除工具,将环境变量LD_PRELOAD设置为在编译之前预加载`差异消除库文件`之后，该库将拦截系统时间及hostname的调用以确保编译状态的一致性。

过程

- 在编译代码之前，设置`差异消除库文件`保存的时间戳及hostname ,然后编译代码。
- 在重建阶段，将`差异消除库文件`设置成相同的时间戳及hostname并进行重建。

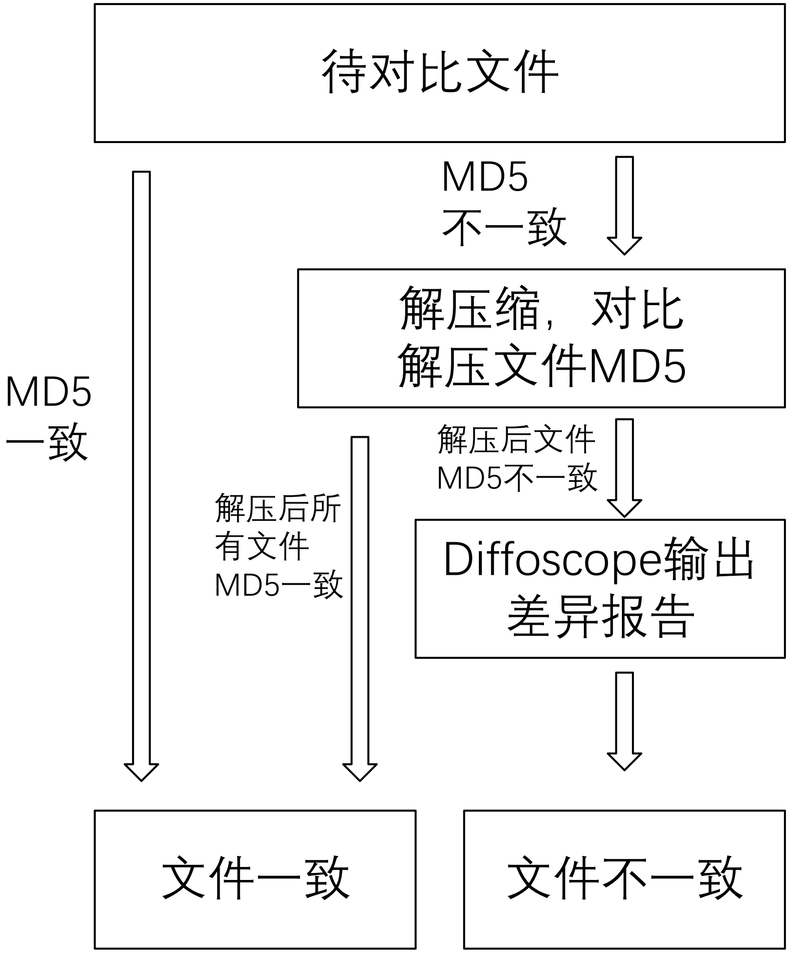
效果

- 产品源代码不需要修改，从而减少了产品在代码纠正方面的投资。
- 商业和开放源代码工具所产生的差异也将得到消除，而无需进行任何代码修改。
- 时间戳的语义被最大程度地保存。
- 该工具支持32位和64位系统。
- 可以通过“ unset LD_PRELOAD”格式的命令禁用该工具。

对libfaketime的扩展

- 黑白名单机制，定制化劫持命令。支持对命令的单函数劫持。
- 劫持hostname命令，消除由于执行机不同导致的BUILDHOST差异

自动化解包对比工具：精准分析文件级差异，差异报告可视化

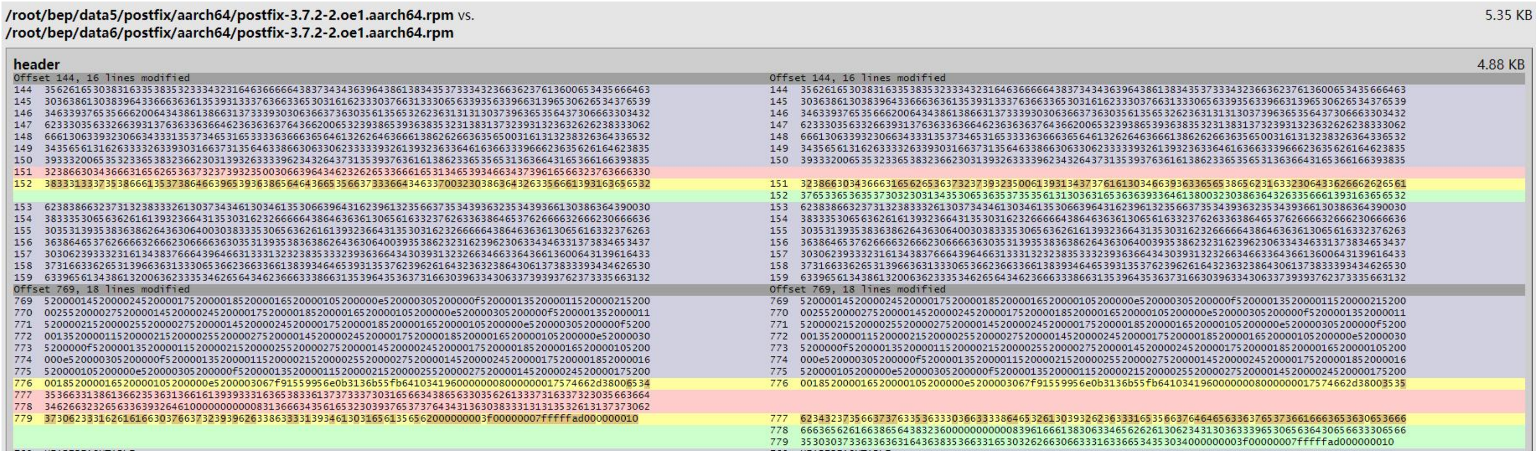


过程

- 对比二进制MD5值，若一样则认为二进制一致
- 二进制MD5不同时，将二进制文件解压，并对两个二进制解压出的对应文件进行MD5比较，若所有文件Md5都一致，则认为二进制一致
- 存在MD5不一致的文件时，使用diffoscope工具输出差异报告，二进制不一致

效果

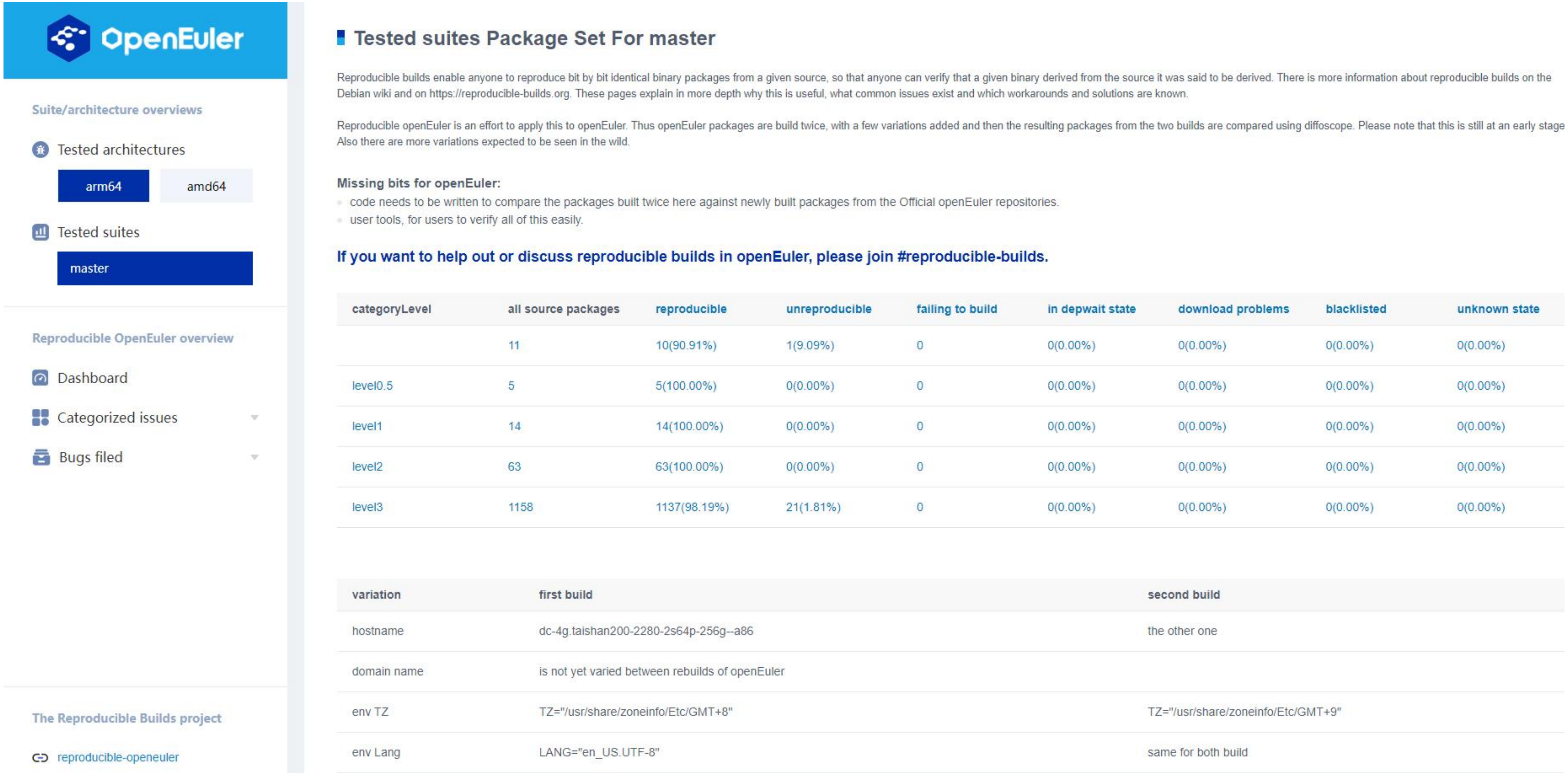
- 可对构建产物进行对比以及解压对比
- 当文件不一致时可输出差异报告
- 对RPM包的RSA，PGP签名文件进行拆分，不进行对比



```
/root/bep/data5/postfix/aarch64/postfix-3.7.2-2.oe1.aarch64.rpm vs.                                    5.35 KB
/root/bep/data6/postfix/aarch64/postfix-3.7.2-2.oe1.aarch64.rpm

header                                                                                                 4.88 KB
Offset 144, 16 lines modified
144  35626165303831633538353233343231646366666438373434363964386138343537333432366362376136006534356666463
145  30363861303839643366636361353931333763663365303161623330376631333065633935633966313965306265343765 39
146  3463393765356662006434386138663137333930306366373630356135653262363131313037396365356437306663303432
147  6233303563326639313763633636646236363637643662006532393865393638353231383137323931323632626238333062
148  6661306339323066343331353734653165333336366636564613262646366613862626636356500316131323832636433653 2
149  3435656131626333326339303166373135646338663063306233333932613932363364616366333966623635626164623835
150  3933320065353233653832366230313932633339623432643731353937636161386233653565313636643165366166393835
151  3238663034366631656265363732373932350030663964346232626533666165313465393466343739616566323763666330
152  3833313337353866613537386466396539363865646436653566373336643463370032303863643263356661393163656532

153  6238386632373132383332613037343461303461353066396431623961323566373534393632353439366130386364390030
154  3833353065636261613932366431353031623266666438646363613065616332376263363864653762666632666230666636
155  3035313935383638626436306400303833353065636261613932366431353031623266666438646363613065616332376263
156  3638646537626666326662306666363035313935383638626436306400393538623231623962306334346331373834653437
157  3030623933323161343837666439646631333132323835333239363664343039313232663466336436613600643139616433
158  3731663362653139663631333065366236633661383934646539313537623962616432363238643061373833393434626530
159  6339656134386132006362333534626564346236663338663135396435363731663039633430633739393762373335663132
Offset 769, 18 lines modified
769  520000145200002452000017520000185200001652000010520000 0e5200003052000 0f5200001352000011520000215200
770  002552000027520000145200002452000017520000185200001652000010520000 0e5200003052000 0f52000013520000 11
771  520000215200002552000027520000145200002452000017520000185200001652000010520000 0e5200003052000 0f5200
772  0013520000115200002152000025520000275200001452000024520000175200001852000016520000105200000e52000030
773  5200000f5200001352000011520000215200002552000027520000145200002452000017520000185200001652000010 5200
774  000e52000030520000 0f520000135200001152000021520000255200002752000014520000245200001752000018520000 16
775  520000105200000e52000030520000 0f52000013520000115200002152000025520000275200001452000024520000175200
776  00185200001652000010520000 0e5200003067f9155995 6e0b3136b55fb641034196000000080000000175 74662d38006534
777  353663313861366235363136616139393331636538336137373337303165663438656330356261333731633732303566 3664
778  3462663232656336393264610000000000000831366634356165323039376537376434313630383331313135326131373730 62
779  373062333162616166303766373239396263386333313934613031656135656200000003f00000007fffffad000000010

Offset 144, 16 lines modified
144  35626165303831633538353233343231646366666438373434363964386138343537333432366362376136006534356666463
145  30363861303839643366636361353931333763663365303161623330376631333065633935633966313965306265343765 39
146  3463393765356662006434386138663137333930306366373630356135653262363131313037396365356437306663303432
147  6233303563326639313763633636646236363637643662006532393865393638353231383137323931323632626238333062
148  6661306339323066343331353734653165333336366636564613262646366613862626636356500316131323832636433653 2
149  3435656131626333326339303166373135646338663063306233333932613932363364616366333966623635626164623835
150  3933320065353233653832366230313932633339623432643731353937636161386233653565313636643165366166393835

151  3238663034366631656265363732373932350061393134373761613034663936333665653865623163323064336266626561
152  3765336536353730323031343530653635373535613130363165363639336461380032303863643263356661393163656532
153  6238386632373132383332613037343461303461353066396431623961323566373534393632353439366130386364390030
154  3833353065636261613932366431353031623266666438646363613065616332376263363864653762666632666230666636
155  3035313935383638626436306400303833353065636261613932366431353031623266666438646363613065616332376263
156  3638646537626666326662306666363035313935383638626436306400393538623231623962306334346331373834653437
157  3030623933323161343837666439646631333132323835333239363664343039313232663466336436613600643139616433
158  3731663362653139663631333065366236633661383934646539313537623962616432363238643061373833393434626530
159  6339656134386132006362333534626564346236663338663135396435363731663039633430633739393762373335663132
Offset 769, 18 lines modified
769  520000145200002452000017520000185200001652000010520000 0e5200003052000 0f5200001352000011520000215200
770  002552000027520000145200002452000017520000185200001652000010520000 0e5200003052000 0f52000013520000 11
771  520000215200002552000027520000145200002452000017520000185200001652000010520000 0e5200003052000 0f5200
772  0013520000115200002152000025520000275200001452000024520000175200001852000016520000105200000e52000030
773  5200000f5200001352000011520000215200002552000027520000145200002452000017520000185200001652000010 5200
774  000e52000030520000 0f520000135200001152000021520000255200002752000014520000245200001752000018520000 16
775  520000105200000e52000030520000 0f52000013520000115200002152000025520000275200001452000024520000175200
776  00185200001652000010520000 0e5200003067f9155995 6e0b3136b55fb641034196000000080000000175 74662d38003535

777  6234323735663737633533633330366333386465326130393262363331653566376464656336376537366166636536306536 66
778  6663656261663865643832360000000000000839616661383063346562626130623431303633396530656364306566333065 66
779  353030373363363631643638353663316530326266306633316336653435303400000003f00000007fffffad000000010
```

openEuler可重复构建看板,加入reproducible-build社区

https://reproducible-builds.openeuler.org/



Tested suites Package Set For master

Reproducible builds enable anyone to reproduce bit by bit identical binary packages from a given source, so that anyone can verify that a given binary derived from the source it was said to be derived. There is more information about reproducible builds on the Debian wiki and on https://reproducible-builds.org. These pages explain in more depth why this is useful, what common issues exist and which workarounds and solutions are known.

Reproducible openEuler is an effort to apply this to openEuler. Thus openEuler packages are build twice, with a few variations added and then the resulting packages from the two builds are compared using diffoscope. Please note that this is still at an early stage Also there are more variations expected to be seen in the wild.

Missing bits for openEuler:

- code needs to be written to compare the packages built twice here against newly built packages from the Official openEuler repositories.
- user tools, for users to verify all of this easily.

If you want to help out or discuss reproducible builds in openEuler, please join #reproducible-builds.

categoryLevel	all source packages	reproducible	unreproducible	failing to build	in depwait state	download problems	blacklisted	unknown state
	11	10(90.91%)	1(9.09%)	0	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)
level0.5	5	5(100.00%)	0(0.00%)	0	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)
level1	14	14(100.00%)	0(0.00%)	0	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)
level2	63	63(100.00%)	0(0.00%)	0	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)
level3	1158	1137(98.19%)	21(1.81%)	0	0(0.00%)	0(0.00%)	0(0.00%)	0(0.00%)

variation	first build	second build
hostname	dc-4g.taishan200-2280-2s64p-256g--a86	the other one
domain name	is not yet varied between rebuilds of openEuler	
env TZ	TZ="/usr/share/zoneinfo/Etc/GMT+8"	TZ="/usr/share/zoneinfo/Etc/GMT+9"
env Lang	LANG="en_US.UTF-8"	same for both build

Roadmap

Epic	目标	Story
openEuler可重复构建（落地）	2022年在openEuler成立可重复构建专项运作、 并得到官方组织reproducible-builds.org认可	整体可重复构建社区运作机制：在可重复构建社区标准章程
		整体工程技术方案：从编译、归档二进制和元数据、到二进制比较、到趋势看板等工具方案
		与官方组织reproducible-builds社区建立合作关系：梳理对官方组织的诉求、工具复用、与组织对我们的要求
		可重复构建客户端工具开发：锁定时间戳、路径、环境、链接顺序、支持各构建系统集成
		2022年openEuler核心软件包代码仓1800+可重复率达成95%
		可重复构建服务端网站参考： https://reproducible-builds.openeuler.org/： 软件列表页面、构建统计页面、每周&每月可重复率趋势图、问题状态页面、自动提交issue给sig组
	2023年全量包达成95%可重复 2024年98%可重复	社区可重复构建SIG组、定义可重构构建规范、各SIG组协同关系、建立推动上游社区修复二进制不一致问题的机制和渠道
		2023年全量包达成95%可重复，可重复构建看板持续优化； 2024年全量包达成98%可重复
		EulerMaker落地可重复构建、验证测试
		构建元数据采集工具：自动生成buildInfo含构建环境、源码、依赖、时间、构建工程参数等， 基于归档构建元数据还原构建环境和构建工程可重复构建
		可重复构建二进制包仓库建设：存储每日构建软件包、和构建元数据buildInfo、支持随时可还原