99 package problems

Kevin Mittman

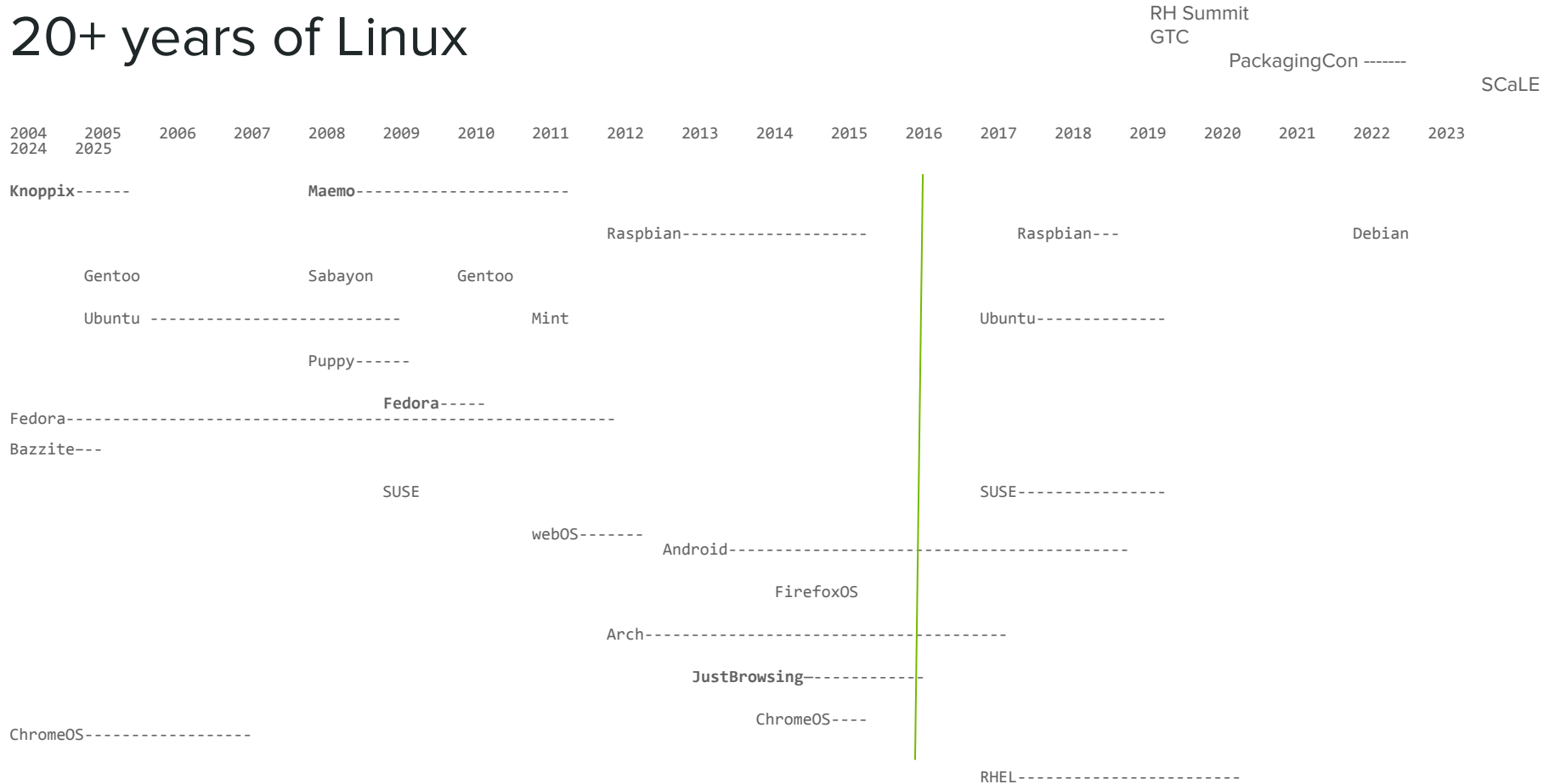# $ whoami

$ whoami

20+ years of Linux

RH Summit
GTC
PackagingCon -------
SCaLE

| 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 | 2017 | 2018 | 2019 | 2020 | 2021 | 2022 | 2023 |
| 2024 | 2025 | | | | | | | | | | | | | | | | | | |

Knoppix------   **Maemo**-----------------------

Raspbian--------------------   Raspbian---   Debian

Gentoo   Sabayon   Gentoo

Ubuntu --------------------------   Mint   Ubuntu-------------

Puppy------

Fedora-----

Fedora--

Bazzite---

SUSE   SUSE----------------

webOS-------   Android--

FirefoxOS

Arch---

JustBrowsing—------------
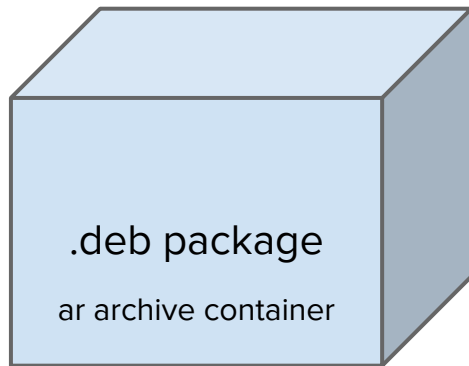
ChromeOS----

ChromeOS------------------
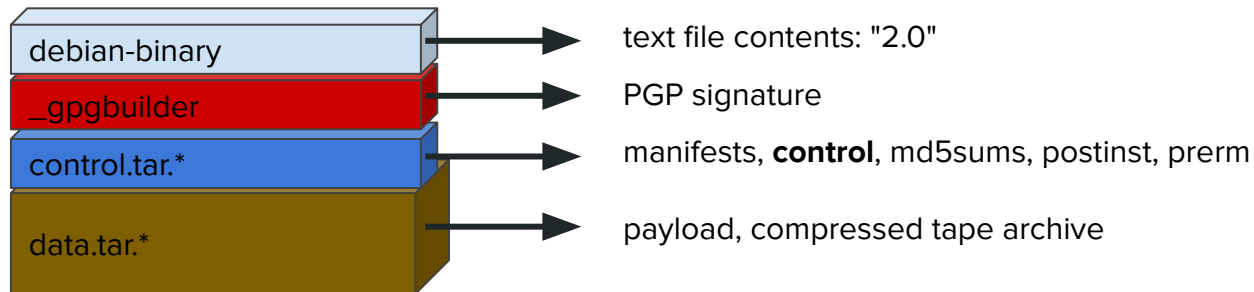
RHEL-----------------------

What are packages

- Common package manager utilities
- Payload files to install (paths and permissions)
- Metadata (package name, version, depends, etc.)
- {pre,post}-install/removal scripts for ldconfig, creating symlinks, or update a DB
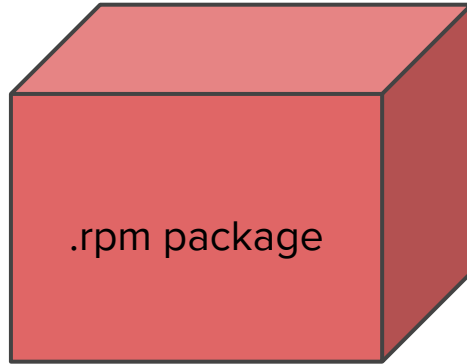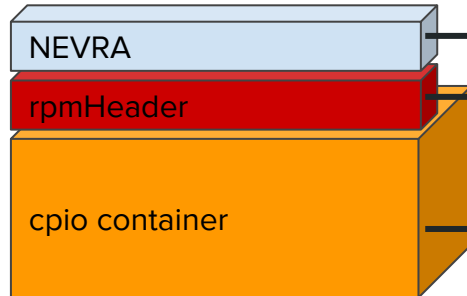
Debian package format


```
ar p *.deb "data.tar.xz" | \
tar --wildcards -xOJf - "*/copyright" | less
```

debian-binary	text file contents: "2.0"
_gpgbuilder	PGP signature
control.tar.*	manifests, **control**, md5sums, postinst, prerm
data.tar.*	payload, compressed tape archive

RPM package format


```
rpm2cpio *.rpm | \
cpio -iv --to-stdout "*/LICENSE" | less
```

NEVRA → Name, Epoch, Version, Release, Arch

rpmHeader → "Magic no." tags: requires, summary,
buildtime, signature

cpio container → payload, cpio archive format
ASCII header

Problems?

- Unsatisfied dependencies
- File conflicts
- Return codes
- Versioning

Debugging

- pkgs.org
- fakeuname
- rpmrebuild
- wiki.archlinux.org/title/Pacman/Rosetta

Debugging

- Debian

```
recurse_apt() { getDeps() { output=$(timeout 5 sudo
apt-get install $1 2>&1 | grep "Depends:");
broken=$(echo "$output" | sed -e 's|but it is
not.*||' -e 's|(.*||' | awk '{print $NF}'); };
getDeps "$1"; while [[ -n $broken ]]; do for dep in
$broken; do echo ":: $dep"; prev="$output"; getDeps
"$dep"; done; done; echo "$prev"; }
```

Debugging

- RPM

```
queryRpm() { __listOption() { local pkgMeta=$(rpm
--nosignature -qp --$2 $1 2>/dev/null | grep -v -E
"(rpmlib|^/)" | paste -sd"," | sed 's|,|, |g'); [[ -n
$pkgMeta ]] && printf "%-11s : %s\n" "${2^}" "$pkgMeta"; };
[[ -f "$1" ]] || return; rpm --nosignature -qp --info "$1"
| head -n 15; list="requires conflicts obsoletes provides
recommends suggests supplements enhances"; for opt in
$list; do __listOption "$1" "$opt"; done; }
```

Debian templates

install file

```
libhello/bin/*          usr/bin/

libhello/lib64/*        usr/lib/x86_64-linux-gnu/

libhello/include/*      usr/include/
```

changelog

```
libhello (1.0.0-1) unstable; urgency=medium

  * Initial release

 -- Name  <email>  $(date -R)
```

compat

12

control file

```
Source: libhello
Maintainer: Name <name@example.com>
Section: libs
Priority: optional

Package: libhello
Section: libs
Architecture: amd64
Depends: ${misc:Depends}
Description: Hello World library
 A library that implements Hello World functionality.
```

rules file

```
#!/usr/bin/make -f
DH_VERBOSE := 1
%:
        dh $@
override_dh_auto_install:
        dh_missing --fail-missing
override_dh_strip:
override_dh_strip_nondeterminism:
override_dh_shlibdeps:
override_dh_dwz:
```

RPM .spec file

```
%global _enable_debug_package 0
%global debug_package %{nil}
%global __os_install_post /usr/lib/rpm/brp-compress
%{nil}

Name: libhello
Version: 1.0.0
Release: 1
Summary: Hello World library

License: Proprietary

Source: libhello-linux-x86_64-%{version}.tar.gz

%description
This is a library that implements Hello World
functionality.

%prep
%setup -q -n libhello
```

```
%install
rm -rf %{buildroot}
mkdir -p %{buildroot}/usr
cp -r bin lib64 include %{buildroot}/usr/

%files
%{_bindir}/hello
%{_libdir}/libhello.so
%{_libdir}/libhello.so.1.0.0
%{_libdir}/libhello_static.a
%{_includedir}/hello.h

%clean
rm -rf %{buildroot}

%changelog
* $(date -R) Name <name@example.com> - 1.0.0-1
- Initial package release
```
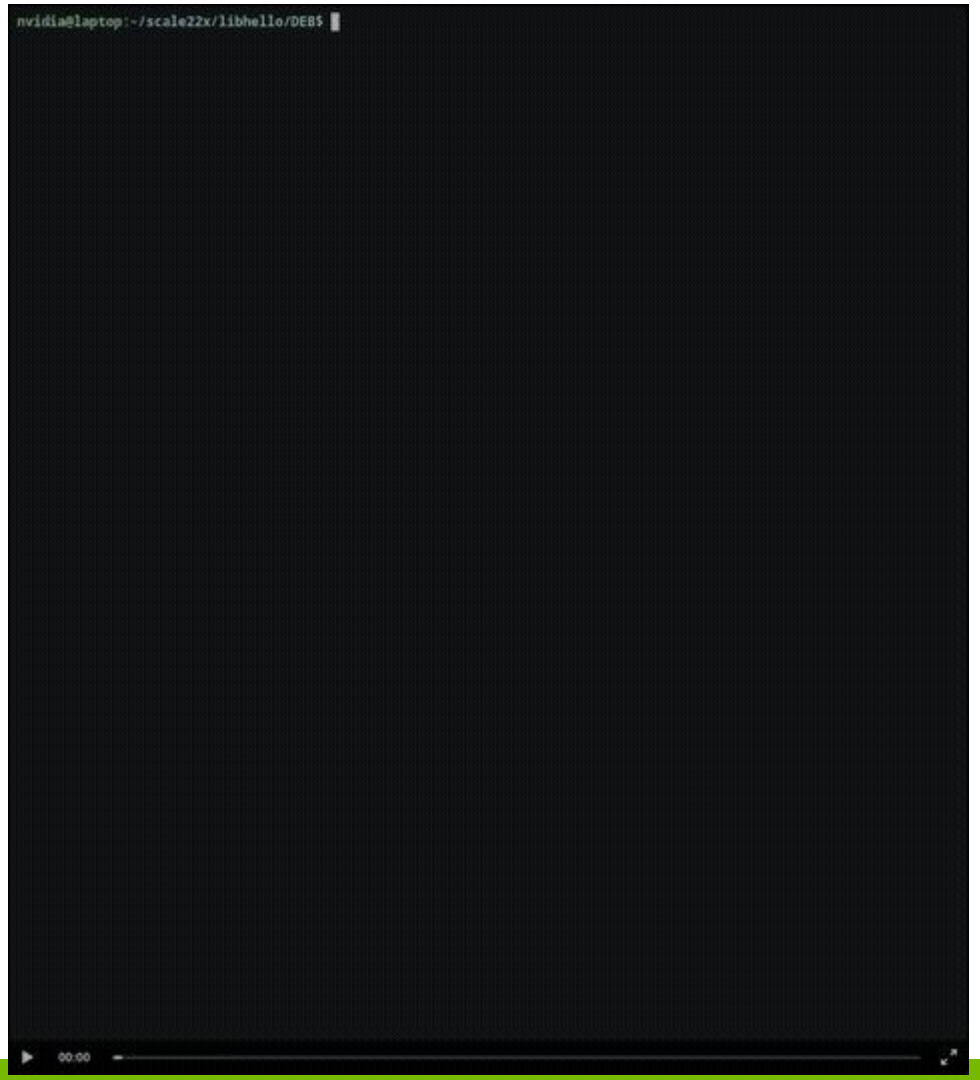
Hello World .deb

dpkg-buildpackage -b

sbuild

```
sudo sbuild-createchroot jammy /srv/schroot
https://archive.ubuntu.com/ubuntu
```

- ```
  sudo debootstrap --arch=amd64 --variant=buildd --verbose
  --include=fakeroot,build-essential --components=main
  --resolve-deps jammy /srv/schroot
  https://archive.ubuntu.com/ubuntu
  ```

```
sbuild -d jammy --add-depends=debhelper libhello-1.0.0
```

Hello World .rpm

```
mkdir -p ~/rpmbuild/{BUILD,
        BUILDROOT,RPMS,SRPMS,
        SOURCES,SPECS}
cp *.spec SPECS/
cp *.tar* SOURCES/
cd ~/rpmbuild
rpmbuild -ba SPECS/*.spec
```

mock

```
mock -r fedora-41-x86_64 --init --bootstrap-chroot

mock -r fedora-41-x86_64 --resultdir=. --buildsrpm --spec *.spec
--sources *.tar*

rpmbuild --rebuild *.src.rpm
```

Versioning

- Fundamentals
  - Versioning is a crucial part of releasing software
  - There is nothing intrinsically special about "1.0" or "9000"
  - Monotonically increasing

Versioning

- Uniqueness
  - Do not re-use version numbers.
  - If there's a problem with your package, bump the release/revision number.
  - Bumping the Epoch value (default: 0) can have unforeseen consequences!

Versioning

- Ecosystem indicators
  - SemVer
  - CalVer
  - Streams or channels

Versioning

- Quantitative comparison
  - `7 < 8 < 9`
  - `5.1 < 5.10`
  - `3.14a =~ 3.14z ?`
  - `1.2.3.4 > 1.2`
  - `400.00, 500.00`

It is important to not simply convert version number "float" strings to integers using multiplication

Versioning

- Character restrictions
  - Debian packages underscore █ characters are field delimiters

    `{name}_{version}-{revision}_{architecture}.deb`

  - RPM packages hyphen ▬ characters are field delimiters
    `{name}-{version}-{release}.{architecture}.rpm`

  - curl utility carat ^ characters are not valid in URLs

  - Regex pattern

    `([0-9]+\.)+[0-9]+`

SHALL WE PLAY A GAME ?

01	02	03	04	05	06
07	08	09	10	11	12
13	14	15	16	17	18

☒1

apt-get install

ERROR: $pkgA : Depends: $pkgB (>= $version) but it is not going to be installed.

Need to follow dependencies to find the broken package (try recurse_apt)

$pkgC : Depends: $pkgD (>= $version) but it is not installable

yum install

file /etc/foo.conf conflicts between attempted installs of %{pkgA}-%{version}-%{release}.%{dist}.%{arch} and %{pkgB}-%{version}-%{release}.%{dist}.%{arch}

Files that install to the same location are ignored, unless the contents are different

zypper install

Problem: installed $pkgA-$x obsoletes $provides <= $version provided by $pkgB-$y
 Solution 1: Following actions will be done:
  downgrade of $pkgA-$x to $pkgA-$y
  deinstallation of $pkgA-$x
  downgrade of $pkgC-$x to $pkgC-$y
  deinstallation of $pkgC-$x
Solution 2: do not install $pkgB-$y
Choose from above solutions by number or cancel [1/2/c] (c):

virtual Provides are messy

04

apt-get install

You might want to run 'apt --fix-broken install' to correct these.

The following packages have unmet dependencies:

  $pkgA : Depends: $pkgC (= $versionA) but $versionB is to be installed

          Depends: $pkgD (= $versionA) but $versionB is to be installed

  $pkgB : Depends: $pkgE (>= $versionC) but it is not going to be installed

E: Unmet dependencies. Try 'apt --fix-broken install' with no packages (or specify a solution)

apt-get install

W: GPG error: _____ InRelease: The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 7ABA1A6E82DB0B48

E: The repository '_____ InRelease' is not signed.

N: Updating from such a repository can't be done securely, and is therefore disabled by default.

GPG pubkey missing

dnf install

Problem: package $pkgA-$version requires $pkgB >= $version, but none of the providers can be installed

- package $pkgA-$version requires $pkgC >= $version, but none of the providers can be installed

- package $pkgB-$version requires $pkgD >= $version, but none of the providers can be installed

- conflicting requests

- nothing provides $pkgE >= $epoch:$version needed by $pkgF-$version

Epoch (re-)set in one of the packages breaking dependencies, requires bumping Epoch to match

apt-get install

dpkg: error: version '-' has bad syntax: revision number is empty

Fix version in changelog file

apt-get install -V $virtualpkg=$version3

Package $virtualpkg is a virtual package provided by:

  $pkgA $version1

  $pkgB $version2

  $pkgC $version3

You should explicitly select one to install.

E: Version '$version3' for '$virtualpkg' was not found

Virtual Provides cannot provide a specific version

09

dpkg --install $pkgA-$version.deb

Unpacking $pkgA ($version) …

dpkg: error processing archive $pkgA-$version.deb

 trying to overwrite '/etc/file', which is also in package $pkgB

Errors were encountered while processing

End-user may have tried to manually delete file but still "owned" by installed $pkgB

Uninstall should include --purge

OR file install path (same or different contents) in two packages

zypper remove $pkgA

(4/10) Removing $pkgA-$version ....[done]

update-alternatives: warning: alternative /usr/lib64/pathA/libfoo_static.a (part of link group libfoo_static.a) doesn't exist; removing from list of alternatives

update-alternatives: warning: /etc/alternatives/libfoo_static.a is dangling; it will be updated with best choice

update-alternatives: using /usr/lib64/pathB/libfoo_static.a to provide /usr/lib64/libfoo_static.a (libfoo_static.a) in auto mode

Typos in update-alternatives

dnf update --best

Problem 1: package $pkgA requires $pkgB >= $version, but none of the providers can be installed

 - package $pkgC-$version conflicts with $pkgB provided by $pkgB-$version

 - cannot install the best update candidate for package $pkgA

(try to add '--allowerasing' to command line to replace conflicting packages or '--skip-broken' to skip uninstallable packages)

Provides was missing for old package name

Follow the Conflicts, Provides, Obsoletes pattern

apt -f install

dpkg: error processing archive $pkgA (--unpack)

 trying to overwrite 'file.conf', which is also in package $pkgB-$version

Errors were encountered while processing: $pkgA

E: Sub-process /usr/bin/dpkg returned an error code (1)

Even identical configs owned by different packages will conflict

OR circular order of dependencies, remove unneeded dependency from $pkgC

apt-get install $pkgA

The following have unmet dependencies:

 $pkgA : Depends: $pkgB (>= $version) but it is not going to be installed

           Depends: $pkgC (>= $version) but it is not going to be installed

E: Unable to correct problem, you have held broken packages.

May depend on packages outside your control (i.e. other repos)

apt-get install

Some packages could not be installed. This may mean that you have requested an impossible situation or if you are using the unstable distribution that some required packages have not yet been created or been moved out of Incoming.

As described there are packages missing from the repository

Warning: seems to use apt-key (provided by apt) without depending on gnupg or gnupg2.

Warning: This will BREAK in the future and should be fixed by the package maintainer(s).

Note: Check first if apt-key functionality is needed at all - it probably isn't!

Warning: apt-key should not be used in scripts (called from postinst maintainerscript of the package)

As described and was removed in Debian 12 and Ubuntu 22.10

16

[MIRROR] $pkgA.rpm: Downloading successful, but checksum doesn't match.
Calculated: %hashB(sha256) Expected: %hashA(sha256)

Re-signing a package will change the checksum

17

Different package/version installed than expected

Multiple packages may Provide the same dependency,

recommend users utilize verbose flag

- zypper -v install
- apt-get install -V
- dnf install (verbose by default)

apt-get install $metapkg

apt-get remove --purge $metapkg

Dependent packages are still installed

Meta-packages are entrypoints, removing does not remove dependents

To remove "orphans" run apt-get --purge autoremove

Thank your package maintainers!

Any Questions?

Installer AMA

GTC code
Mar 17 - 21