Welcome to LUG

sh.lug.umbc.edu/checkin


UMBC LINUX USERS GROUP

Please Check In:

sh.lug.umbc.edu/checkin

Club Updates:

- Coming Up:
  - Elections (April 29th) - All positions can be run for - No incumbents running for Treasurer.
    - Add Google Form here
  - MeshPages Presentation - Devon (May 6th)
  - SAS workshop and server upgrade (May 13th / Study Day)
  - Summer Break!

Packages and Package Managers

What is a system without software?

nothing :(

How to ins

- Building fr
  - Pull sourc
  - Compile th
  - Add to you
  - Configure

There’s gotta

Packages!!

Package Archives

- A package is a **packaged** piece of software
- They are the files needed to install a piece of software onto a system

Packages are distributed as package **archives**.

- An archive is just a set of files in a container. (Not always compressed)

Package archives have two components:

- **Program Files** - Files needed to install program (Not always pre-compiled)
- **Package metadata** - more on this later

Package ArchivesFormats

No system like this is useful without some standardization

Some Examples

- Deb
- RPM
- .msi
- APK
- .pkg.tar.zst
- AppImage
- Flatpak
- Snap
- PUP & PET

Flatpak

.deb Debian Package Format

- A standard package format developed for debian
- Used by it's derivative operating systems as well

.deb File Structure

> **pkg.deb** - (UNIX ar Archive - Predecessor to TAR)
>
> > **debian-binary** - txt file containing ver # e.g. '2.0'
> >
> > **control.tar** - pkg meta information
> >
> > **data.tar** - install files

```
control.tar.xz
data.tar.xz
debian-binary
```

.deb Debian Package Format

Within **control.tar**

```
root@code-server:~/nano/control# ls
conffiles  control  md5sums  postinst  prerm
```

- control - description and dependencies
- md5sums - checksums of all files in package
- conffiles - defines the data files that are configs and shouldn't be overwritten
- preinst, postinst, prerm, postrm - optional script files
- config - optional script for debconf (could include picture)
- shlibs - shared library dependencies

.deb Debian Package Format

Control File

- **Pkg name**
- **Version**
- **Architecture**
- **Maintainer**
- **Size**
- **Dependenceies**
- **Conflicts**
- **etc..**

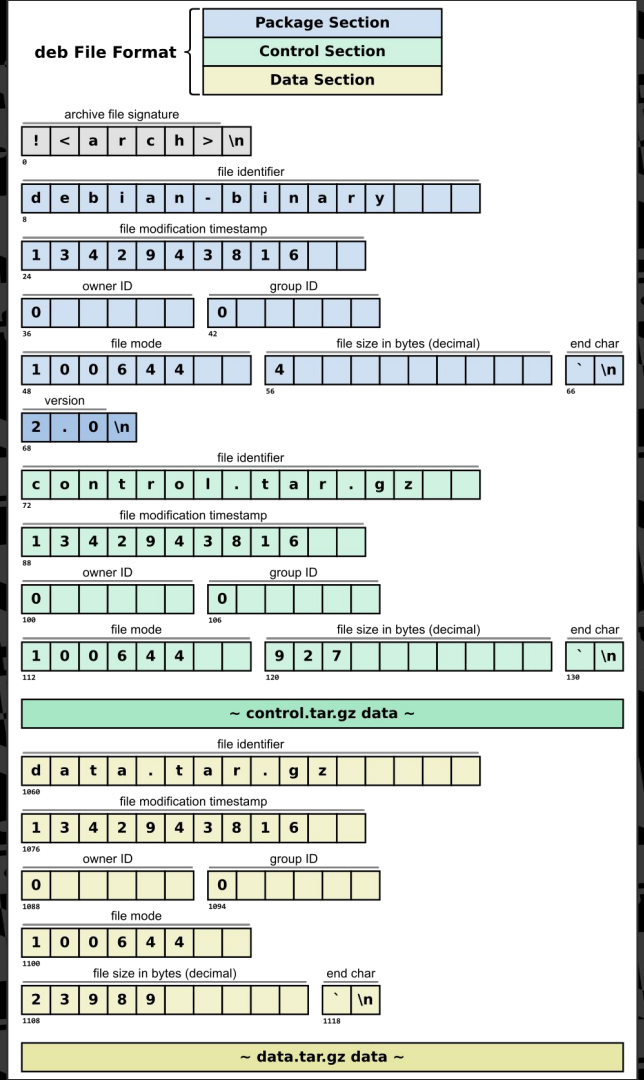
```
Package: nano
Version: 8.7.1-1
Architecture: amd64
Maintainer: Jordi Mallach <jordi@debian.org>
Installed-Size: 2922
Depends: libc6 (>= 2.38), libncursesw6 (>= 6), libtinfo6 (>= 6)
Suggests: hunspell
Conflicts: pico
Breaks: nano-tiny (<< 2.8.6-2)
Replaces: nano-tiny (<< 2.8.6-2), pico
Section: editors
Priority: important
Homepage: https://www.nano-editor.org/
Description: small, friendly text editor inspired by Pico
 GNU nano is an easy-to-use text editor originally designed as a replacement
 for Pico, the ncurses-based editor from the non-free mailer package Pine
 (itself now available under the Apache License as Alpine).
 .
 However, GNU nano also implements many features missing in Pico, including:
  - undo/redo
  - line numbering
  - syntax coloring
  - soft-wrapping of overlong lines
  - selecting text by holding Shift
  - interactive search and replace (with regular expression support)
  - a go-to line (and column) command
  - support for multiple file buffers
  - auto-indentation
  - tab completion of filenames and search terms
  - toggling features while running
  - and full internationalization support
```

.deb File Structure

Thanks to wikimedia for this goated diagram

.deb files support signature validation for security. Not commonly implemented.

Instead debian does something else…



So we have packages
How do we manage them?

package managers :)

Package Managers

Debian uses APT (Advanced Package Tool)

- Pacman and DNF are prevalent as well

APT is really just a high-level wrapper for DPKG.

- DPKG is the low level package manager for debian

DPKG - Debian Package

Low level package manager

Responsible for:

- Installing packages `dpkg -i nano.deb`
- Uninstalling packages `dpkg -r nano`
- Configuring packages
- Managing what's installed `dpkg -l`

Has it's limitations

Which brings us back to…

APT - Advanced Package Tool

Higher level DPKG wrapper

- Has functions that call DPKG
- Can install and uninstall packages

What makes APT different

- Resolves and installs dependences when installing packages
- Can retrieve packages from remote sources

How does APT work?

- Pulls from remote sources (repositories) as defined in **/etc/apt/sources.list** and **/etc/apt/sources.list.d**
  - These sources can be remote URI's e.g. **http://mirror.lug.umbc.edu/debian**
  - Can also be CD repositories

```
root@code-server:/etc/apt# cat sources.list
#deb cdrom:[Debian GNU/Linux 12.7.0 _Bookworm_ - Official amd64 NETINST with firmware 20240831-10:38]/ bo
okworm contrib main non-free-firmware

deb http://deb.debian.org/debian/ bookworm main non-free-firmware
deb-src http://deb.debian.org/debian/ bookworm main non-free-firmware

deb http://security.debian.org/debian-security bookworm-security main non-free-firmware
deb-src http://security.debian.org/debian-security bookworm-security main non-free-firmware
```

Debian repository structure

See it for yourself: https://mirror.lug.umbc.edu/debian

> **dists/** - contains all the releases like; trixie, stable

> **dists/bookworm/** - contains sub components and then architectures

- Indexed by **release** file containing hashes of every **Packages.gz file**

> **dists/bookworm/main/binary-amd64/Packages.gz** - Actual index of package files for this release. Contains hash information and file path

How does APT work?

- Pulls from remote sources (repositories) as defined in **/etc/apt/sources.list** and **/etc/apt/sources.list.d**
  - These sources can be remote URI's e.g. **http://mirror.lug.umbc.edu/debian**
  - Can also be CD repositories

```
root@code-server:/etc/apt# cat sources.list
#deb cdrom:[Debian GNU/Linux 12.7.0 _Bookworm_ - Official amd64 NETINST with firmware 20240831-10:38]/ bo
okworm contrib main non-free-fi

deb http://deb.debian.org/debian/ bookworm main non-free-firmware
deb-src http://deb.debian.org/debian/ bookworm main non-free-firmware

deb http://security.debian.org/debian-security bookworm-security main non-free-firmware
deb-src http://security.debian.org/debian-security bookworm-security main non-free-firmware
```

Debian repository structure

Release File

```
Origin: Debian
Label: Debian
Suite: oldstable
Version: 12.13
Codename: bookworm
Changelogs: https://metadata.ftp-master.debian.org/changelogs/@CHANGEPATH@_changelog
Date: Sat, 10 Jan 2026 22:28:35 UTC
Acquire-By-Hash: yes
No-Support-for-Architecture-all: Packages
Architectures: all amd64 arm64 armel armhf i386 mips64el mipsel ppc64el s390x
Components: main contrib non-free-firmware non-free
Description: Debian 12.13 Released 10 January 2026
MD5Sum:
 0ed6d4c8891eb86358b94bb35d9e4da4  1484322 contrib/Contents-all
 d0a0325a97c42fd5f66a8c3e29bcea64    98581 contrib/Contents-all.gz
 1ed4eee5ea09e2dfae1997ef5c8290ce   840868 contrib/Contents-amd64
 b4b844bb746b52088b449383c07f028e    61033 contrib/Contents-amd64.gz
 b6b9802ee1e267db32434f1c24118cfa   479536 contrib/Contents-arm64
...
 ade79acd45fd8d963a749bb88b46134c   231032 contrib/binary-amd64/Packages
 d9be8f9d1a38fa3fa90f89cdd0c902b9    64763 contrib/binary-amd64/Packages.gz
 27991041c8abcff62cfecb47e719728e    53480 contrib/binary-amd64/Packages.xz
 36a29e34a163ed88e372dd055005ed7a      123 contrib/binary-amd64/Release
 646d0f8e2164cc69166aa0b865b7014f   194417 contrib/binary-arm64/Packages
 3a78cecedc4d0fb34de81d46846aa0f7    54527 contrib/binary-arm64/Packages.gz
 93d5677924fff41bd3fdb46132bbb0f4    45664 contrib/binary-arm64/Packages.xz
 850afa2f00a01e5bb8c6c9c678d0ad35      123 contrib/binary-arm64/Release
 7936ac7101faa5a633813549df9d94d7   164024 contrib/binary-armel/Packages
```

Debian repository structure

Packages.gz File

```
Package: python3-lib389
Source: 389-ds-base
Version: 2.3.1+dfsg1-1+deb12u1
Installed-Size: 2268
Maintainer: Debian FreeIPA Team <pkg-freeipa-devel@alioth-lists.debian.net>
Architecture: all
Replaces: 389-ds-base (<< 1.4.0.18-1~), python-lib389 (<< 1.3.7.8)
Depends: python3-argcomplete, python3-argparse-manpage, python3-dateutil, python3-distro, python3-ldap, p
ython3-pkg-resources, python3-pyasn1, python3-pyasn1-modules, python3:any, libnss3-tools, openssl, python
3-packaging, python3-pytest
Conflicts: 389-ds-base (<< 1.4.0.18-1~), python-lib389 (<< 1.3.7.8)
Description: Python3 module for accessing and configuring the 389 Directory Server
Homepage: https://directory.fedoraproject.org
Description-md5: 312a10ddcf41c03aed17c8e2759b4410
Section: net
Priority: optional
Filename: pool/main/3/389-ds-base/python3-lib389_2.3.1+dfsg1-1+deb12u1_all.deb
Size: 386208
MD5sum: ff9e4ce45e6e7b86e52fff8373f28ebd
SHA256: c1428eb014c6a02b94e1e294e202a995d7a5a75b429aea5a2c95bd066325cb40
```

How does APT work?

> **apt update**

- Synchronizes **Packages.gz** indices with all remote sources

> **apt upgrade**

- Compares local package versions with that of the index; then pulls and installs needed packages

> **apt install pkg-name**

- Searches releases > Pulls package archive > Installs
  - Recursive for installing dependencies

How does APT validate authenticity?

> When **updating** APT verifies **Packages.gz** integrity against checksum in **release** file

> When **pulling** APT verifies package archive (pkg.deb) against checksum in **Packages.gz**

> When **installing** DPKG verifies install files against local md5sums in package archive.

Trust is passed down

Trusting the source

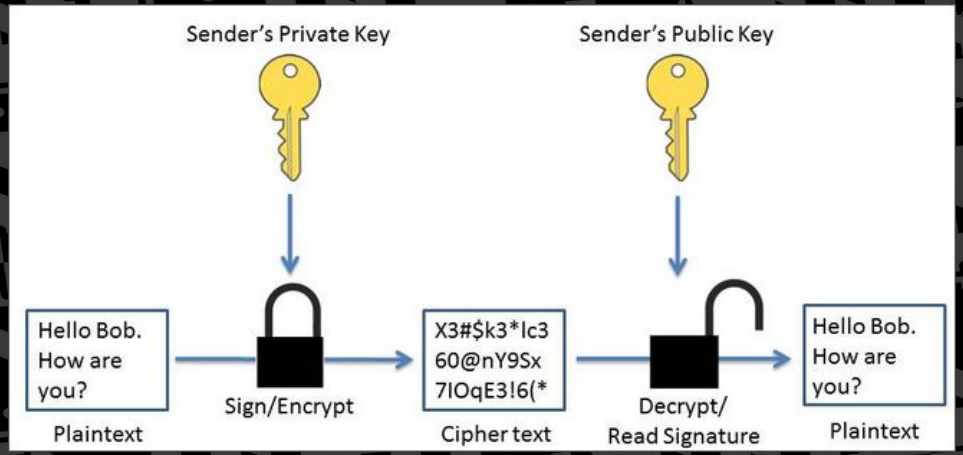
Everything in this process relies on the **release** file at the root of the release directory being authentic.

The Solution:

Release file gets signed with public key encryption.

Hash of **release** file encrypted with private key and stored as **release.gpg**

Clients decrypt with pub key and compare against their own hash of **release**



Trusting the source

Where do you get the public key?

> Often distributed with the OS

> Keys are stored in **/etc/apt/trusted.gpg.d**

You can add keys to rust new repositories

```
root@code-server:/etc/apt/trusted.gpg.d# ls -la
total 92
drwxr-xr-x 2 root root  4096 Apr 21 20:30 .
drwxr-xr-x 9 root root  4096 Sep 12  2024 ..
-rw-r--r-- 1 root root 11861 Jul 30  2023 debian-archive-bookworm-automatic.asc
-rw-r--r-- 1 root root 11873 Jul 30  2023 debian-archive-bookworm-security-automatic.asc
-rw-r--r-- 1 root root   461 Jul 30  2023 debian-archive-bookworm-stable.asc
-rw-r--r-- 1 root root 11861 Jul 30  2023 debian-archive-bullseye-automatic.asc
-rw-r--r-- 1 root root 11873 Jul 30  2023 debian-archive-bullseye-security-automatic.asc
-rw-r--r-- 1 root root  3403 Jul 30  2023 debian-archive-bullseye-stable.asc
-rw-r--r-- 1 root root 11861 Apr  9  2025 debian-archive-trixie-automatic.asc
-rw-r--r-- 1 root root 11873 Apr  9  2025 debian-archive-trixie-security-automatic.asc
-rw-r--r-- 1 root root  1384 Apr  9  2025 debian-archive-trixie-stable.asc
```

Other Package Managers

Most package managers follow similar concepts to APT/DPKG

Some notably different package managers / formats

- **Pacman (AUR)** - Built for uncompiled packages
- **Flatpak, Snap & AppImage**
  - Built to run on any system
  - Achieved by bundling all dependencies in one container
  - Consequently bloated, heavy and inefficient