System maintenance

(Redirected from Backup)

Regular system maintenance is necessary for the proper functioning of Arch over a period of time. Timely maintenance is a practice many users get accustomed to.

Related articles

General recommendations

1 Check for errors

1.1 Failed systemd services

Check if any systemd services have failed:

```
$ systemctl --failed
```

See **systemd#Using units** for more information.

1.2 Log files

Look for errors in the log files located in `/var/log/`, as well as messages logged in the systemd journal:

```
# journalctl -b
```

See **systemd/Journal** for more information and filtering options.

See **Xorg#Troubleshooting** for information on where and how **Xorg** logs errors.

2 Backup

Having backups of important data is a necessary measure to take, since human and machine processing errors are very likely to generate corruption as time passes, and also the physical media where the data is stored is inevitably destined to fail.

See **Synchronization and backup programs** for many alternative applications that may better suit your case. See **Category:System recovery** for other articles of interest.

It is encouraged to automate backups, see **Autostarting#On time events**.

2.1 Configuration files

Before editing any configuration files, create a backup so that you can revert to a working version in case of problems. Editors like **vim** and **emacs** can do this automatically. On a larger scale, consider using a **configuration manager**.

For **dotfiles** (configuration files in the home directory), see **dotfiles#Tracking dotfiles directly with Git**.

2.2 List of installed packages

Maintain a list of all installed packages so that if a complete re-installation is inevitable, it is easier to re-create the original environment.

See **pacman tips#List of installed packages** for details.

2.3 Pacman database

See **pacman/Tips and tricks#Back up the pacman database**.

2.4 Encryption metadata

See **Data-at-rest encryption#Backup for disk encryption scenarios**.

2.5 System and user data

See **System backup**.

3 Upgrading the system

It is recommended to perform full system upgrades regularly via **pacman#Upgrading packages**, to enjoy both the latest bug fixes and security updates, and also to avoid having to deal with too many package upgrades that require manual intervention at once. When requesting support from the community, it will usually be assumed that the system is up to date.

Make sure to have the Arch install media or another Linux "live" CD/USB available so you can easily rescue your system if there is a problem after updating. If you are running Arch in a production environment, or cannot afford downtime for any reason, test changes to configuration files, as well as updates to software packages, on a non-critical duplicate system first. Then, if no problems arise, roll out the changes to the production system.

If the system has packages from the **AUR**, carefully upgrade all of them.

pacman is a powerful package management tool, but it does not attempt to handle all corner cases. Users must be vigilant and take responsibility for maintaining their own system.

3.1 Read before upgrading the system

Before upgrading, users are expected to visit the **Arch Linux home page (https://archlinux.org/)** to check the latest news, or alternatively subscribe to the **RSS feed (https://archlinux.org/feeds/news/)** or the **arch-announce mailing list (https://lists.archlinux.org/mailman3/lists/arch-announce.lists.archlinux.org/)**. When updates require out-of-the-ordinary user intervention (more than what can be handled simply by following the instructions given by *pacman*), an appropriate news post will be made.

Before upgrading fundamental software (such as the **kernel**, **xorg**, **systemd**, or **`glibc` (https://archlinux.org/packages/?name=glibc)**) to a new version, look over the appropriate **forum (https://bbs.archlinux.org/)** to see if there have been any reported problems.

Users must equally be aware that upgrading packages can raise **unexpected** problems that could need immediate intervention; therefore, it is discouraged to upgrade a stable system shortly before it is required for carrying out an important task. Instead, wait to upgrade until there is enough time available to resolve any post-upgrade issues.

Tip: You could use a pacman hook like **`informant` (https://aur.archlinux.org/packages/informant/)** AUR which prevents you from updating if there is fresh Arch News that you have not read since the last update ran.

3.2 Avoid certain pacman commands

Avoid doing **partial upgrades**. In other words, never run `pacman -Sy`; instead, always use `pacman -Syu`.

Generally avoid using the `--overwrite` option with pacman. The `--overwrite` option takes an argument containing a glob. When used, pacman will bypass file conflict checks for files that match the glob. In a properly maintained system, it should only be used when explicitly recommended by the Arch Developers. See the **#Read before upgrading the system** section.

Avoid using the `-d` option with pacman. `pacman -Rdd package` skips dependency checks during package removal. As a result, a package providing a critical dependency could be removed, resulting in a broken system.

3.3 Partial upgrades are unsupported

Arch Linux is a **rolling release** distribution. That means when new **library** versions are pushed to the repositories, the **Developers (https://archlinux.org/people/developers/)** and **Package Maintainers** rebuild all the packages in the repositories that need to be rebuilt against the libraries. For example, if two packages depend on the same library, upgrading only one package might also upgrade the library (as a dependency), which might then break the other package which depends on an older version of the library.

That is why partial upgrades are **not supported**. Do **not** use:

- `pacman -Sy package`
- `pacman -Sy` followed by `pacman -S package` (Note the absence of `-Su` in the installation of the package.)
- `pacman -Syuw` (Note that `pacman -Syuw` does imply the same risks like `pacman -Sy`, as it will update the *pacman* sync database without installing the newer packages.)

When refreshing the package database, **always** do a full upgrade with `pacman -Syu`. Note that if `pacman -Syu` does not perform the upgrade because of an error, the end result is the same as running `pacman -Sy`. Therefore, the error must be resolved and the upgrade operation completed as soon as possible.

Be very careful when using `IgnorePkg` and `IgnoreGroup` for the same reason. If the system has locally built packages (such as **AUR** packages), users will need to rebuild them when their dependencies receive a **soname** bump.

If a partial upgrade scenario has been created, and binaries are broken because they cannot find the libraries they are linked against, **do not "fix" the problem simply by symlinking**. Libraries receive **soname** bumps when they are **not backwards compatible**. A simple `pacman -Syu` to a properly synced mirror will fix the problem as long as *pacman* is not broken.

The bash script *checkupdates*, included with the **`pacman-contrib (https://archlinux.org/packages/?name=pacman-contrib)`** package, provides a safe way to check for upgrades to installed packages without running a system update at the same time, and provides an option to download the pending updates to the pacman cache without touching the sync database.

3.4 Act on alerts during an upgrade

When upgrading the system, be sure to pay attention to the alert notices provided by **pacman**. If any additional actions are required by the user, be sure to take care of them right away. If a pacman alert is confusing, search the forums and the recent news posts for more detailed instructions.

3.5 Deal promptly with new configuration files

When *pacman* is invoked, *.pacnew* and *.pacsave* files can be created. Pacman provides notice when this happens and users must deal with these files promptly. Users are referred to the pacman/Pacnew and Pacsave wiki page for detailed instructions.

Also, think about other configuration files you may have copied or created. If a package had an example configuration that you copied to your home directory, check to see if a new one has been created.

3.6 Restart or reboot after upgrades

Upgrades are typically not applied to existing processes. You must restart processes to fully apply the upgrade.

The archlinux-contrib (https://archlinux.org/packages/?name=archlinux-contrib) package provides a script called checkservices (https://github.com/archlinux/contrib/blob/master/admin/checkservices) which runs pacdiff to merge *.pacnew* files then checks for processes running with outdated libraries and prompts the user if they want them to be restarted.

The kernel is particularly difficult to patch without a reboot. A reboot is always the most secure option, but if this is very inconvenient kernel live patching can be used to apply upgrades without a reboot.

3.7 Revert broken updates

If a package update is expected/known to cause problems, packagers will ensure that pacman displays an appropriate message when the package is updated. If experiencing trouble after an update, double-check pacman's output by looking at `/var/log/pacman.log`.

Tip: You can use a log viewer such as wat-git (https://aur.archlinux.org/packages/wat-git/)AUR to search the pacman logs.

At this point, only after ensuring there is no information available through pacman, there is no relevant news on https://archlinux.org/, and there are no forum posts regarding the update, consider seeking help on the forum (https://bbs.archlinux.org), over IRC, or by downgrading the offending package.

3.8 Check for orphans and dropped packages

After upgrading you may now have packages that are no longer needed or that are no longer in the official repositories.

Use `pacman -Qtd` to check for packages that were installed as a dependency but now, no other packages depend on them. If an orphaned package is still needed, it is recommended to change the installation reason to explicit. Otherwise, if the package is no longer needed, it can be removed. See pacman/Tips and tricks#Removing unused packages (orphans) for details.

Additionally, some packages may no longer be in the remote repositories, but they still may be on your local system. To list all foreign packages use `pacman -Qm`. Note that this list will include packages that have been installed manually (e.g., from the AUR). To exclude packages that are (still) available on the AUR, use the script from BBS#288205 (https://bbs.archlinux.org/viewtopic.php?id=288205#p2116629) or try the ancient-packages (https://aur.archlinux.org/packages/ancient-packages/)AUR tool.

4 Use the package manager to install software

Pacman does a much better job than you at keeping track of files. If you install things manually you *will*, sooner or later, forget what you did, forget where you installed to, install conflicting software, install to the wrong locations, etc.

- Install packages from the official repositories using the method in the pacman#Installing packages section.

- If the program you desire is not available, check to see if someone has created a package in the AUR. Follow the method in that article for installation.
- Lastly, if the program you want is not in the official repositories or in the AUR, learn how to create a package for it.

To clean up improperly installed files, see pacman/Tips and tricks#Identify files not owned by any package.

4.1 Choose open-source drivers

Always try open source drivers before resorting to proprietary drivers. Most of the time, open source drivers are more stable and reliable than proprietary drivers. Open source driver bugs are fixed more easily and quickly. While proprietary drivers can offer more features and capabilities, this can come at the cost of stability. To avoid this dilemma, try to choose hardware components known to have mature open source driver support with full features. Information about hardware with open source Linux drivers is available at linux-drivers.org (https://web.archive.org/web/20190819193521/http://www.linux-drivers.org/).

4.2 Be careful with unofficial packages

Use precaution when using packages from the AUR or an unofficial user repository. Most are supplied by regular users and thus may not have the same standards as those in the official repositories. Avoid AUR helpers which automate installation of AUR packages. **Always** check PKGBUILDs for sanity and signs of mistake or malicious code before building and/or installing the package.

To simplify maintenance, limit the amount of unofficial packages used. Make periodic checks on which are in actual use, and remove (or replace with their official counterparts) any others. See pacman/Tips and tricks#Maintenance for useful commands. Following system upgrade, use `rebuild-detector (https://archlinux.org/packages/?name=rebuild-detector)` to identify any unofficial packages that may need to be rebuilt.

4.3 Update the mirrorlist

Update pacman's mirrorlist, as the quality of mirrors can vary over time, and some might go offline or their download rate might degrade.

See mirrors for details.

5 Clean the filesystem

When looking for files to remove, it is important to find the files that take up the most disk space. Programs that help with this are found in:

- List of applications/Utilities#Disk usage display.
- List of applications/Utilities#Disk cleaning.

5.1 Package cache

Remove unwanted *.pkg* files from `/var/cache/pacman/pkg/` to free up disk space.

See pacman#Cleaning the package cache for more information.

5.2 Unused packages (orphans)

Remove unused packages from the system to free up disk space and simplify maintenance.

See **pacman/Tips and tricks#Removing unused packages (orphans)** for details.

5.3 Old configuration files

Old configuration files may conflict with newer software versions, or corrupt over time. Remove unneeded configurations periodically, particularly in your home directory and `~/.config`. For similar reasons, be careful when sharing home directories between installations.

Look for the following directories:

- `~/.config/` -- where applications stores their configuration
- `~/.cache/` -- cache of some programs may grow in size
- `~/.local/share/` -- old files may be lying there

See **XDG Base Directory support** for more information.

To keep the home directory clean from temporary files created at the wrong place, it is a good idea to manage a list of unwanted files and remove them regularly, for example with **rmshit.py (https://github.com/lahwaacz/Scripts/blob/master/rmshit.py)**.

`rmlint (https://archlinux.org/packages/?name=rmlint)` can be used to find and optionally remove duplicate files, empty files, recursive empty directories and broken symlinks.

5.4 Broken symlinks

Old, broken symbolic links might be sitting around your system; you should remove them. Examples on achieving this can be found **here (https://unix.stackexchange.com/questions/34248/how-can-i-find-broken-symlinks)** and **here (https://www.commandlinefu.com/commands/view/8260/find-broken-symlinks)**. However, you should not blindly delete all broken symbolic links, as some of them serve a purpose **[1] (https://unix.stackexchange.com/questions/151763/is-there-a-downside-to-deleting-all-of-the-broken-symbolic-links-in-a-system)**.

To quickly list all the broken symlinks of your system, use:

```
# find / -xtype l -print
```

Then inspect and remove unnecessary entries from this list.

6 Tips and tricks

The following tips are generally not required, but certain users may find them useful.

6.1 Use proven software packages

Arch's rolling releases can be a boon for users who want to try the latest features and get upstream updates as soon as possible, but they can also make system maintenance more difficult. To simplify maintenance and improve stability, try to avoid cutting edge software and install only mature and proven software. Such packages are less likely to receive difficult upgrades such as major configuration changes or feature removals. Prefer software that has a strong and active development community, as well as a high number of competent users, in order to simplify support in the event of a problem.

Avoid any use of the testing repository, even individual packages from testing. These packages are experimental and not suitable for a stable system. Similarly, avoid packages which are built directly from upstream development sources. These are usually found in the **AUR**, with names including things like: "dev", "devel", "svn", "cvs", "git", etc.

6.2 Install the linux-lts package

The **`linux-lts (https://archlinux.org/packages/?name=linux-lts)`** package is an alternative Arch kernel package, and is available in the **core** repository. This particular kernel version has long-term support (LTS) from upstream, including security and bug fixes. It is useful if you use out-of-tree kernel modules and want to ensure their compatibility or if you want a fallback kernel in case a new kernel version causes problems.

To make it available as a boot option, you will need to update your **boot loader**'s configuration file to use the LTS kernel and ram disk: `vmlinuz-linux-lts` and `initramfs-linux-lts.img`.

7 See also

- **Arch News Bash Script (https://bbs.archlinux.org/viewtopic.php?id=146850)**
- **fwupd**

Retrieved from "https://wiki.archlinux.org/index.php?title=System_maintenance&oldid=819943#Backup"