Paquets Universels

AppImage vs Flatpak vs Snap

Sébastien Golouboff Bartosz Tulisz

Table des matières

1 L'intérêt des paquets universels

Dans l'écosystème Linux, la distribution des programmes est gérée par les mainteneurs des différentes distributions, qui s'occupent de créer des versions pour de multiples architectures et même parfois de corriger certains bugs. Le problème de cette approche, appelée *downstream packaging*, quelques inconvénients qui peuvent poser problème pour la distribution d'applications sur Linux.

La diversité de distributions vient avec une diversité de gestionnaires et de formats de paquets incompatibles entre eux. Pour rendre une application facilement installable sur une certaine distribution, il faut donc soliciter ses mainteneurs ou créer et maintenir un paquet au format adéquat, tout en prenant en compte la politique de *packaging*.

Rendre une application facilement accessible sur de nombreuses distributions Linux peut donc rapidement devenir une tâche ardue pour une équipe de développement, et il serait plus simple pour elle d'effectuer un minimum de maintenance tout en maximisant le nombre de distributions supportées par leur application.

Plusieurs format de paquets universels (c'est-à-dire pouvant être installés sur des distributions différentes) ont été développés pour répondre à cette problématique.

2 AppImage

2.1 Présentation

AppImage est un format de fichier paru en 2004 sous licence libre MIT, écrit en C par Simon Peter et maintenu par une communauté de développeur. Il sert à distribuer des applications Linux sous la forme d'un fichier unique contenant toutes les ressources nécessaires à l'exécution de l'application.

2.2 Structure d'une AppImage

Une AppImage est un fichier au format ELF, le format standard des fichiers exécutables Linux. Elle est séparée en deux parties principales embarquées à l'intérieur.
D'abord, un système de fichier contenant :

- Un **point d'entrée** (nommé AppRun) qui est un fichier lançant l'application. Il peut s'agir d'un fichier binaire, un script ou tout autre type de fichier pouvant être exécuté.

- Un fichier **.desktop** et d'une **icône** qui identifient l'AppImage et peuvent être utilisés pour l'intégrer au bureau de l'utilisateur.

- Un **répertoire** (nommé AppDir) contenant toutes les ressources et les dépendances de l'application.

Ce système de fichier est compressé au format SquashFS, un système de fichiers compressé en lecture seul. SquashFS réduit la taille du système de fichiers notamment en compressant les fichiers et en réduisant la taille des répertoires et des i-noeuds à 8 bits.

La seconde partie essentielle de l'AppImage est son *runtime*. Il s'agit de sa partie exécutable qui, à l'exécution, a pour rôle de décompresser et monter le système de fichier à un emplacement temporaire, et d'exécuter le point d'entrée. Il contient également des utilitaires qui permettent par exemple d'extraire le contenu de l'AppImage ou d'uniquement monter le système de fichiers, qui peuvent être invoqués en lui passant certaines options.

2.3 Avantages

Le principal avantage des fichiers AppImage est l'absence de prérequis pour les télécharger et les lancer. En effet, il suffit seulement de télécharger une AppImage mise à disposition par les développeurs d'une application et de la rendre exécutable pour la lancer, ce qui contribue à faciliter l'expérience des utilisateurs.

Par rapport à Flatpak et Snap, les AppImage sont à la fois plus légère et plus rapide car contrairement à ces deux autres formats de paquets universels, une AppImage ne s'exécute pas dans un conteneur.

2.4 Inconvénients

Les AppImage sont plus volumineuses que les paquets Linux classiques. En effet, toutes les librairies et dépendances de l'application sont incluses dans le répertoire AppDir. Cela est néanmoins mitigé par la compression au format SquashFS.

Il faut également noter que les AppImages ne disposent pas des avantages de la conteneurisation dont profitent Snap et Flatpak, même si cela peut être résolu en exécutant les AppImages dans un environnement "bac-à-sable" comme Firejail.

3 Flatpak

3.1 Présentation

Le Gestionnaire de paquets Flatpak, de son ancien nom **xdg-app**, est développé par Alexander Larsson depuis Août 2007 dans le langage C.
Flatpak est disponible sous près de 33 distributions Linux et préinstallés sous certaines comme Fedora ou Linux Mint. Il est comme AppImage un projet Open Source.

3.2 Fonctionnement

Voici les différentes étapes exécutées par Flatpak à la création d'un paquet :
Le répertoire build est créé si il n'existe pas déja.
Le code source des différents modules est téléchargé et vérifié, puis installé et monté. On apn rajoute le build au dépôt distant si il est spécifié.

3.3 Un paquet Flatpack

Une archive Flatpak à pour extension .flatpakref et contient :
— *Le runtime* : Un ensemble de librairies minimales servant à lancer l'application (partagés entre les Flatpaks)
— *Le SDK* : Qui contient le compilateur et le déboggueur.
— L'ensemble des dépendances nécessaires à l'application.
— Un Manifeste, un fichier Yaml contenant les informations sur le Flatpak (le Runtime, SDK, dépendances utilisées ...).
— Le ou les fichiers sources de l'application.

3.4 Points Positifs

Ces éléments sont isolés du système et de ces ressources, ainsi si l'application nécessite d'accéder aux ressources utilisateurs (fichiers, ...) une fenêtre s'ouvre demandant à l'utilisateur l'autorisation ou non d'y accéder. Flatpak étant un gestionnaire de paquets, il permet la mis à jour des paquets puisqu'il se base sur un système de branches pour gérer les différents dépôts distants contenant les applications installées (à l'image de Git). On peut ainsi faire tourner plusieurs versions d'un même paquet en parallèle et disposer de plusieurs branches d'un Flatpak (stable, unstable, ...). Le Flathub, store applicatif de Flatpak, regroupe les applications disponibles sous ce format de paquet. Les paquets Flatpak peuvent être gérés pas différents utilitaires graphiques (GNOME logiciels, KDE Discover) pour les utilisateurs ne souhaitant pas passer par les lignes de commandes.

3.5 Points négatifs

L'isolation proposée par Flatpak a un côut en taille puisqu'il est nécessaire de fournir à la création du paquet l'ensemble des dépendances requises pour l'application. On peut ainsi se retrouver avec plusieurs fois la même librairie d'installée pour des applications différentes. Les Runtimes eux, qui sont des ensembles de librairies sont partagés entre les différents Flatpaks.

4 Snap

4.1 Présentation

Snap est un gestionnaire de paquets développé par la société Canonical (éditrice d'Ubuntu) et est disponible sous près de 19 distributions linux mais également sous Mac OS et Windows (en preview). Snap repose sur la fonctionnalité Systemd qui n'est pas présente sous certaines distrubutions comme FreeBsd par exemple.

Snap se décompose en plusieurs utilitaires :

— Snapd : Le gestionnaire de paquets snap
— Snapcraft : Permettant la création de paquets et l'export vers les dépôts distants
— Snap qui est le nom donné à un paquet avec ce gestionnaire.

4.2 Fonctionnement

Comme AppImage, une archive snap est au format SquashFS contenant les librairies nécessaires, le manifest pour monter l'application, le Runtime et les fichiers binaires. On retrouve beaucoup de points communs entre Flatpak et Snap, les deux gestionnaires implémentent le sand-boxing même si Snap permet de moduler l'isolation système en spécifiant dans le manifest du paquet quel niveau d'isolation appliquer (Strict/Classic/DevMode). Snap permet aussi de gérer plusieurs branches d'une même application (Stable / unstable/ ...).

4.3 Points Positifs

Il dispose de son propre store applicatif, snapcraft.io. Snapd vérifie régulièrement si des mises à jour sont disponibles pour les snaps installés et les appliquent en arrière plan et de manière atomique grâce à l'encodage delta. C'est à dire que seulement les fichiers modifiés sont téléchargés.

4.4 Points négatifs

Un paquet Snap est de taille similaire à un paquet Flatpak, on est donc sur une taille 2 à 3 fois supérieure à celle d'une AppImage Les mises à jour automatiques ne sont pas désactivables, bien que l'on puisse définir une fréquence de mises à jour. On note de plus un temps de lancement assez conséquent.

5 Benchmark

Le Benchmark ci-dessous compare les 3 gestionnaires de paquets pour l'application LibreOffice 6.2 sous Ubuntu.

On peut constater que AppImage est le plus rapide et le plus léger, bien qu'en contrepartie il n'assure pas l'isolation de l'application avec le système.

- Benchmark de LibreOffice 6.2 sur Ubuntu

	Flatptak	Snapcraft	AppImage
Taille paquet	519MB	543MB	248MB
Temps de lancement	7s	13s	3s

6 Un échec d'adoption ?

Malgré leur âge (plus de 15 ans), on constate que les paquets sont encore loin de l'adoption massive espérée par leur créateurs.

Nous pensons que cela peut être dû, d'une part à un échec de résoudre le problème de la fragmentation de l'écosystème Linux. En effet, il existe maintenant plusieurs formats de paquets universels qui ne sont pas supportés de manière égale par toutes les applications.

D'autre part, le modèle d'*upstream packaging* peut être considéré comme antithétique à la manière dont sont maintenus les distributions Linux. En effet, ce sont les mainteneurs de celle-ci qui effectuent un travail d'"harmonisation" et d'"assainissement" des dépôts en amont en appliquant des correctifs et en distribuant les logiciels dans des versions qui permettent la meilleur interopérabilité possible. C'est ce même principe de *downstream packaging* qui apporte une garantie de stabilité à des distributions populaires comme Debian ou Ubuntu.

7 Bibliographie

Cliquez sur un titre pour suivre le lien :
— *Flatpak*, Site du gestionnaire de paquets Flatpak.
— *Flathub*, Store applicatif de Flatpak.
— *The Flatpak Security Model*, Alexander Larsson.
— *Snap*, Site du gestionnaire de paquets Snap.
— *Benchmark De Libre Office 6.2 sur Ubuntu*
— *Différences snap Vs Flatpak Vs AppImage*