Flatpak: Easy, Fast, Safe

Christian Hergert
@hergertme
What is Flatpak?
What is Flatpak?

- A distribution mechanism to **efficiently** and **safely** ship bits to users
- A robust sandbox to **protect** users from intrusive applications
- **Decentralized** design to avoid app-store lock-in
- A suite of **developer tooling** to simplify application composition
- Ship releases and bug fixes **more frequently** and with **less latency**
Who are we?
Who makes Flatpak?

• Inception by Alex Larsson of the GNOME Project
  • Informed by lessons learned implementing Glick, Glick 2, and fixing file-system plumbing in the docker project
  • Colin Walters built these great technologies called OSTree (git for operating systems) and linux-user-chroot (stronger sandboxing) now bubblewrap
  • Red Hat, Endless, Collabora, Codethink, Intel, Kinvolk, Solus, and an ever growing list of individual contributors
Flatpak provides to users...
Easy installation and update

- **Cross distribution** from day 0
- **GNOME Software** supports Flatpak
  - Used on many distributions and desktops outside of GNOME
  - Buy-in from other Free Software projects like Solus, KDE, and more
- **Command line tools** for people who prefer them
- Native support for **proprietary graphics drivers**
  - Lots of coordination to make this happen such as *statically linking GL drivers* from vendors and Mesa
  - Even supports ancient kernels like mainline Linux 3.2
Efficient installs and updates

• Most people still have relatively slow or partially connected internet
  • Failure to accept this is exclusionary
• Static deltas provide efficient release-to-release downloads
  • Combination of single file download and bsdiff for tight updates
• Zero-to-installed is also a static delta for new installations
• Without static deltas, efficient git-style tree compare-and-sync
• Application vs Runtime split reduces download size when multiple applications are installed
Trustworthy applications

• Application meta-data is **cryptographically verified** in depth
  • Compare this to git-SHA1 which does not verify tree in depth
• Applications **updated atomically**, either they succeed or no change is made
• Applications can safely **update while running**
  • This often breaks with distributions, where 99% of the time it works, but 1% of the time your system is left with inconsistent state
• Rigid **sand-boxing** with **Portals** for elevating privilege via **safe** API
• Applications will continue to **run for years** and **across OS upgrades**
• Strong integration with **Wayland** security model
• Apps cannot snoop on each other (or even know they are installed)
Flatpak provides to devs...
Cross-distribution

- The first packing system **designed from ground-up** to be cross-distribution
  - Doesn’t rely on ABI of host-based libraries
  - Doesn’t rely on out-of-tree kernel LSM, supports **SELinux**
  - Uses **mainline Linux** kernel features, suid helper for older kernels
- Only requires POSIX compliant file-system
  - Hard-link farms, **content addressing**, potential for btrfs/xfs **relink**
  - **Atomic upgrade**, even while application is running
- **Runtimes** provide **predictable** and **reliable user-space**
Robust build tooling

- **flatpak-builder** wrangles together dependencies, patches, and your app
- OSTree cache-points for unreasonably **fast partial rebuilds**
- Getting **closer to reproducible builds** (not there yet, but closer)
  - Turns out sharing compilers and predictable build runtime helps a lot
- Control over dependencies which have been **Q/A tested**
- Build and test in the **same environment** as your users
- IDE integration with **Builder**
- Profiler integration with **Sysprof**
- **Debugger** integration coming to Builder 3.26
Runtimes and SDKs

- Allows projects to share common libraries
- Allows for smaller per-app downloads
- **Shared burden** for CVE tracking
- You probably shouldn’t make your own
  - Reuse Freedesktop, **GNOME**, KDE, etc
- An SDK is a runtime without “developer” bits removed
  - Headers, debug symbols, compilers, associated tooling, etc
- Your app can rely on an SDK too, **Builder** targets **org.gnome.Sdk**
Safety-focused Portals

- Portals run **out of process**, app does not get raw access
- Document portal seamlessly gives access to $HOME
  - **FUSE/fd-pass** to grant app ability to read/write
- Open documents, URLs, etc with installed applications
- “Capture” portal to **take a photo**
- We’re **enhancing** daemons like **PulseAudio** and **Piños** for sand-boxing
- Plenty more to write! (Come join us!)
Get more testing

• Support for concurrent application channels
  • Stable, Beta, Nightly, etc
• You test the same runtime environment as your users
• Multiple architecture support
  • With modern QEMU and Linux kernel, you can run ARM on x86_64!
Flatpak provides distributions...
Help

• Distributions are **universally overworked**

• Compete on what you’re good at, **building the OS** rather than Sisyphean tasks like app packaging

• Many applications relying on a few number of runtimes could allow us to reduce CVE tracking and patching load for all distributions
  • This could mean that your applications gets **CVE updates faster** than it otherwise would thanks to shared ownership over runtimes

• Applications that bundle security related components with vulnerabilities is still a concern, but mitigated through robust sand-boxing

• Automated CVE tracking can be a major win for developers (on my ToDo list)
Improve OS security

- Robust sand-boxing to **protect your users** from third-party apps
- **Improve your security** story, by sand-boxing apps that would otherwise be shipped without sand-boxing
- Support for the in-tree LSM, **SELinux**
- **D-Bus Filtering** means applications can’t communicate with each other
- Xorg is not suitable for both security and efficient graphics
  - We can discuss the design issues at the core of the Xorg protocols and currently available extensions *(See me afterwards if you’d like to)*
• Many libraries **break ABI in very subtle ways**
  • ABI is nebulous once you move past symbols and structure layout
  • We’ve tried to fix upstreams for years, unsuccessfully
  • Turns out it’s really hard and those that tell you otherwise might not understand the problem fully
• Even worse in the Node, Python, Ruby, Go, and other H-L-L communities
  • They lack the rigid adherence to SONAME and ABI version semantics that are more common in the C (and sometimes C++) communities
  • Python applications often break due to shared Python package dependency that breaks API between releases
Reduce hosting costs

• Smaller full-build ISOs
• Remove application distribution costs from your bandwidth
• If you host a flatpak repository you get small, incremental updates
• Your mirrors can update more often and faster
Demo Time!
Thank You!