# The Future of Flatpak

Sebastian Wick (swick)
fosstodon.org/@swick
sebastian.wick@redhat.com

LAS

# Status

- More and more Flatpaks
- Flathub is doing great
- Distributions are starting to pre-install Flatpak Apps
- More and more Portals
- Everything is great

# Status

- More and more Flatpaks
- Flathub is doing great
- Distributions are starting to pre-install Flatpak Apps
- More and more Portals
- Everything is great...?

# Status

- Flatpak itself is stagnant!
- Maintenance and security work is happening
- Feature PRs just sit there for month and years
- Maintainers have left the project
- Newcomers don't get the opportunity to get familiar with the code base, get feedback, etc.
- Vicious cycle which leads to stagnation

# Status

Maybe this is fine...?

# Status

Flatpak is stable!

# Status

How to improve the ecosystem,
without changing Flatpak itself too much

# Flatpak Improvements: Pre-installing

- Distros want to install Flatpak Apps as part of the Base OS
- Distro/admin configures the Apps, flatpak-preinstall takes care of the rest
- The feature is implemented (Kalev Lember, Owen Taylor, me) and will be in RHEL10
- Sitting in a pull request...

# Flatpak Improvements: OCI image/transport

- Remotes are a single ostree repo
- Repos of the size of Flathub become problematic
- ostree is uncommon, has bespoke, non-standard tooling
- Building Flatpaks also requires non-standard tooling

# Flatpak Improvements: OCI image/transport

- Flatpak supports pulling OCI images from container registries
- Images get imported locally into the ostree repo
- Gives us standard OCI registry tooling
- Gives us standard OCI build tooling
- zstd:chunked OCI transport means we retain file-level de-duplication (same as ostree)
- There are a few PRs which should improve things around OCI support

# Flatpak Improvements: Backwards Compatible Permissions

- Continued push towards more restricted permissions and portals
- For example `--device=input` replaces `--device=all`, the USB portal replaces `--device=all`
- Flatpak apps can't drop `--device=all` because some systems are stuck on an old Flatpak or Portal versions
- Need a backwards compatible way to describe things!

# Flatpak Improvements: Backwards Compatible Permissions

- `--device=input --device=all --nodevice-if=all:has-input-device`
- `--device=all --nodevice-if=all:has-usb-portal`
- This also replace the ad-hoc wayland/X11 fallback mechanism:

  `--socket=wayland --socket=x11`

  `--nosocket-if=x11:has-wayland`
- This feature exists in a pull request

# Flatpak Improvements: cgroup

- Currently for every running instance, a directory in ***`$XDG_RUNTIME_DIR/.flatpak`*** is created with metadat
- Inside the mount namespace, ***`/.flatpak-info`*** is created, so that ***`/proc/$PID/root/.flatpak-info`*** can be used to authenticate a process as a Flatpak  instance
- Flatpak tells systemd to move instances to an appropriate cgroup

  … but this is allowed to fail.
- We can require a cgroup per instance (either via systemd or directly) and store more metadata in the xattrs of the cgroup

# Flatpak Improvements: PipeWire

- Sound is still routed through PulseAudio, even when the host uses PipeWire
- With PulseAudio, speakers and microphones are bundled together: you get either both or none
- Pipewire can expose a restricted set of nodes
- Flatpak could tell PipeWire to restrict nodes on connections from the cgroup of the Flatpak instance to speakers only
- Mount the host PipeWire socket, just like we do for Wayland
- Portals could be used to dynamically add/remove nodes

# Flatpak Improvements: Nested sandboxes

- Flatpak currently doesn't allow nested user namespaces (turned off via seccomp)
- Instead, a session helper service can be called to create a "side-sandbox"
- This mostly works, but it is fragile and there are issues which will likely not get solved
- Nested user namespaces are turned off, because
  - It increases the attack surface against the kernel
  - Portals need to authenticate Flatpak instances via `/proc/$PID/root/.flatpak-info`, but mount namespaces can override this

# Flatpak Improvements: Nested sandboxes

- User namespaces have matured, making the increased attack surface argument less strong
- Making it easy for apps to drop privileges increases the overall security
- User namespaces allow us to drop a bunch of complexity
- Using cgroups, we can drop the requirement that `/proc/$PID/root/.flatpak-info` has to describe the instance
- With that in place, we might be able to enable nested sandboxes!

# Flatpak Improvements: xdg-dbus-proxy

- Flatpak spawns a xdg-dbus-proxy for every Flatpak instance
- Apps can only talk with the proxy, not the session bus directly
- xdg-dbus-proxy is responsible for filtering according to rules that Flatpak sets up on `flatpak-run`
- The rules start from a `deny-all` state, and allow-list specific names
  - This is done because services might expose things other apps are not supposed to use

# Flatpak Improvements: xdg-dbus-proxy

- We should move the filtering from xdg-dbus-proxy to dbus brokers directly
- Policy based on a cgroup path
- Going to work on a prototype in busd

# Flatpak Improvements: xdg-dbus-proxy

- Should allow for a more dynamic policy where apps can export services to other apps
  - Aside: Apps can already communicate with other apps

    The network namespace is shared and there are tons of side-channels if one tries hard enough
- A portal invocation by the app goes app → proxy → broker → portal → broker → backend → broker → portal → broker → proxy → app
- We could get rid of two processes in the chain by removing the proxy

# Flatpak Improvements: Network namespace

- Currently flatpak runs the app in the host network namespace
- Abstract unix sockets, TCP and UDP services, all leak into the sandboxes
- For example, the AusweisApp (German ID card authentication App) exposes a service on localhost, which is available to all apps
- The problem isn't that communication between Apps is inherently bad, but that Apps accidentally expose more than they should to other Apps
- We should fix this, and create network namespaces!
- Please reach out to us if you have experience!

# Flatpak Improvements: Drivers

- GL/Vulkan drivers get built against the Flatpak runtime
- Multiple drivers for multiple runtime versions
- Storage and network traffic overhead
- When a runtime is EOL, drivers are not updated
  - Unmaintained apps will not support current GPUs
  - Can mean the app won't work anymore

# Flatpak Improvements: Drivers

- Simple idea: use the host drivers
- Valve uses libcapsule to get the host driver into the runtime, but it's fragile
- Statically compile drivers and use them on the host, in containers and in Flatpak
- Mount latest glibc and static host drivers into runtimes

# Portals

- A lot of “Flatpak problems” are not really Flatpak problems at all, but missing Portals
- Improving Portals directly improves the Flatpak story

# Portals
# Documents

- The document portal has some problems and no one working on it
- Fine grained permissions via the document portal do not work for all Apps
- Some apps want to manage a whole “library”, potentially on removable media, multiple directories, …
  - Blender, Steam, Music players, …
- A library portal could dynamically bind-mounts user-selected host locations into the sandbox

# Portals
# The Rest

- There are lots of ideas for Portals!
- AutoFill/password, FIDO, hidraw, restore, speech synthesis, AI, timers, ...
- Portals are kind of hard to write: written in C, threading, async, ...
- Currently exploring making it easier by using fibers (libdex) to replace threads and async/callback based C code
- Might even make sense to rewrite it in Rust

# Flatpak Next

If we wanted to
replace **Flatpak** with a **Flatpak Next**,
how would that look like?

# Flatpak Next

- Flatpaks are normal OCI images, built by generic OCI tooling
- Flatpaks get distributed via generic OCI registries
- std:zchunked and composefs to retain file-level de-duplication
- Clients have a local flatpak container storage
- Some files from images get exported (desktop file, service, etc.)
- flatpak-run creates the namespace directly (bwrap was only useful for systems without user namespaces)
- flatpak-run creates composefs mount from OCI images (app + runtime) and mounts an overlayfs on top for config

# Flatpak Next

# The Future of Flatpak

So, What Does the Future of Flatpak look like?

# The Future of Flatpak

So, What Does the Future of Flatpak look like?

Incremental improvements

Alignment with the wider container ecosystem

Sebastian Wick (swick)
fosstodon.org/@swick
sebastian.wick@redhat.com