Sandboxed Applications for GNOME

GUADEC 2013

Lennart Poettering

August 2013
Who we are
Our goal: We want GNOME to be the modern, general purpose OS
And “Apps” are a crucial part of it
Apps =

sandboxed user applications, shipped in single file per app, no privileges for execution, stable ABIs, reliable testability
Apps =
sandboxed user applications,
shipped in single file per app,
no privileges for execution,
stable ABIs,
reliable testability
RPMs/DEBs =

RPMs/DEBs =
installable only by root,
live in a common namespace,
vendor APIs,
huge test matrix
We want both, RPMs/DEBs for building the system, and sandboxed user apps to run on top of it.
RPMs/DEBs: primarily focussed around distributions as single provider, builder, tester of programs

Apps: many sources from the internet, untrusted code
Apps

Key feature: isolation from the surrounding OS and user private data

For security reasons

And for API stability testability/building reasons

(But not everywhere: think extensions)
We want kernel-level isolation
We want a free, community-based, vendor-agnostic solution
9 Steps
1 – Make kdbus work, so that we can have kernel-enforced bus sandboxes, and so that we can use it to transfer major data in and out of the sandbox via the bus.
2 – App sandboxes build on Linux namespaces, seccomp, cgroups, capabilities.
2 – App sandboxes build on Linux namespaces, seccomp, cgroups, capabilities.

(2.5 – Stricter File Hierarchy Specification)
3 – Introduce *Portals* infrastructure as primary way in and out of the sandbox for applications. Portals are an interactive security scheme that doubles as integration technology.
4 – App images as compressed file systems with multiple partitions in a loopback file, one for each architecture plus a common base set.
5 – An extended search path logic
In GLib and friends
6 – A sandbox aware display manager

Wayland
7 – A apps-aware configuration scheme
dconf
8 – A system for building apps
Profiles
9 – App stores, by any community or vendor
That’s all, folks!