



Box Office

Stephan Bergmann

September 2015

On packaging LibreOffice as a sandboxed xdg-app bundle

xdg-app

[https://wiki.gnome.org/Projects/SandboxedApps:](https://wiki.gnome.org/Projects/SandboxedApps)

- “There are two main goals with this project.
 - “We want to make it possible for 3rd parties to create and distribute applications that work on multiple distributions.
 - “We want to run the applications with as little access as possible to the host. (For example user files or network access.)”

LibreOffice

The code base we all love and fear:

- ~300 configure switches
 - From `--enable-debug` to `--with-lang` to `--with-system-zlib`
- ~100 external submodules
 - From `boost` to `python3` to `librevenge`
- ~100 localizations

LibreOffice

In other words:

- LibreOffice is an ideal testbed to try out all kinds of technology
- So lets get LibreOffice working inside xdg-app!



The API that isn't

- For Mac OS X and Windows, it is natural to have TDF provide LO builds for users
- Linux is fractured
 - rpm vs. deb, different library SONAMEs, ...
 - Each distro packages LO themselves
- But sometimes users want to download fresh versions from TDF
 - If only to do testing on them
- TDF builds for Linux settle on a lowest common denominator baseline
 - No GTK3
 - Old GnomeVFS instead of GIO/GVFS (and what about KDE users?)
 - ...

Package once, run everywhere

- xdg-app defines a precise environment an app can expect at runtime
- Will let TDF finally build just one version of LO for Linux
 - Without paying attention to an outdated baseline's requirements

bibisect

The bibisect git repos are typically built by volunteers with fat machines

- Sometimes dependencies on the volunteer's environment sneak in
- Frustrating for others wanting to use the repo
- But re-building repos would be expensive, too

Building LibreOffice for xdg-app

- SDK based on the somewhat obscure Yocto project
 - No Perl Archive::Zip module, needed during LO build
 - No GLU (removed the few uses from LO code base)
 - xml-config hardcoded as “exit 1”
- --disable-cups, -gconf, -gltf, -gstreamer-1-0, -orcus, -postgresql_sdbc
- --without-java
- --with-system-libs, but ~30 --without-system-* overrides
- X11-based, for now

- ~350M installation tree

Sandboxing

- Simpler packaging of LibreOffice is good
- Improved security via sandboxed execution is even better
 - Reduce risk when viewing documents obtained from strangers
 - User not trusting provider of LibreOffice probably less of a concern
- xdg-app makes that easy to opt-in/-out
 - `xdg-app run --filesystem=host`

Sandboxed File Access

- Main issue for an office suite is granting access to read/write files outside the sandbox
- The intuitive model is “If the user explicitly specifies a file outside the sandbox, then grant access to it from inside the sandbox”
 - “File › Open...” delegating to the ContentPortal D-Bus service
 - `xdg-app run org.libreoffice.LibreOffice ~/Documents/text.odt`

LibreOffice Peculiarities

- Simple apps may be fine with a pathname or even an open fd to access a file
- LibreOffice expects to have more control
 - `~lock.*#` files
 - Containing user information presented when attempting concurrent access
 - Can even contain multiple entries for special multi-user features
- But LibreOffice knows to distinguish between (presumed local) file: and other URL schemes
- xdg-app API still in flux
 - GVFS-based document: URL scheme would have fit LibreOffice nicely
 - Now at a FUSE-/pathname-based approach

Bits and Pieces

Some open issues:

- Careful to have only one instance running at a time for any per-user configuration data
 - First instance listens on a Unix-domain socket, and other instances send their command lines there and immediately exit
- What to do with the ~100 localizations
 - (plus offline help content)
- Have more of the remaining ~30 dependencies of LibreOffice addressed in an xdg-app runtime
 - Java?

“I want a certificate that allows me to make as big a box office as possible” –Ridley Scott