Sandboxing all programs by default

energizer **Jun 2020**

Linux desktop applications are relatively unconstrained by default: they can access the user's home directory, execute any program, and interact with arbitrary network locations. This free access is convenient, but also raises security concerns around misconfigured or misbehaving programs.

There are several existing tools to help with application security.

- **Fedora Silverblue** is an immutable Linux distribution that distributes packages using the Flatpak format, which runs applications in a **Flatpak application sandbox** by default.
- Since 2005, Linux has had a "secure computing" feature, **`seccomp`** , which allows restricting applications' access to various resources. **Firejail** is a program that takes advantage of `seccomp` to sandbox applications, limiting their access to files and interfaces that the user allows. Firejail also provides a **large set of configurations** for popular applications, allowing these applications the resources they need and denying the resources they don't need.
- Another tool is **Bubblewrap** , which says:

> bubblewrap works by creating a new, completely empty, mount namespace where the root is on a tmpfs that is invisible from the host, and will be automatically cleaned up when the last process exits.

There has been **some** prior **discussion** of application sandboxing in NixOS, so it sounds like people are interested. I'm interested in continuing the conversation, focusing on how we can move toward a more comprehensive solution that enables convenient but safe application usage *by default*.

For example, would it be realistic to create a version of nixpkgs that wraps *all* application binaries in a firejail/flatpack/bubblewrap sandbox execution environment? If not, what would be a better path forward?

Some way to achieve opt-out (as opposed to opt-in) app sandboxing

How many people are paid to work on Nix/Nixpkgs?

aanderse **Jun 2020**

Skip to main content

Maybe @peterhoeg has some thoughts on `firejail`...

8573 **Jun 2020**

Another technology that may be relevant is **AppArmor** . Currently this is used to ‘armor’ `ping` and a small number of services. Apparently it is used somewhat more broadly in Ubuntu; I hadn’t thought until now to look up **their AppArmor profiles** , but, doing so, I see that they have profiles for some important targets that NixOS doesn’t seem to be AppArmor-ing, such as Web browsers and some widely-used services.

---

ajs124 **Jun 2020**

I have something based on AppArmor similar to **systemd confinement** . It only works for systemd services though and is sadly not in a state fit for nixpkgs. It’s been on my todo list to clean it up and post it somewhere, but I’m not sure when I’ll get around to it.

There are even plans (or rather ideas) to expand it for other apps, maybe make it system wide.

In general, for all these approaches, `nix` can give you the closure of any storepath through `exportReferencesGraph`/`closureInfo`. So for path based stuff, you can generate profiles from that with an additional whitelist for paths etc. the application needs.

---

aszlig **Jun 2020**

> ajs124:
>
> I have something based on AppArmor similar to **systemd confinement**. It only works for systemd services though and is sadly not in a state fit for nixpkgs. It’s been on my todo list to clean it up and post it somewhere, but I’m not sure when I’ll get around to it.

I’ve also written something **like this** a while ago for packaging proprietary games. What it basically does is setting up mount/user/pid/uts/ipc namespaces and bind-mount all the runtime dependencies at **build time** (seeing that comment by @Profpatsch reminds me that I should probably replace it by **`closureInfo`**), other dependencies from **`PATH`-like variables** and **extra paths** .

The implementation at the moment isn’t as locked down as I’d want it for a more generic

Skip to main content mple it allows access to the X server), but that’s because - as

mentioned - it was built **for games** (examples: **1** **2**) after all.

Is your implementation public somewhere?

8573 **Aug 2020**

Some recent work on AppArmor in NixOS is **https://github.com/NixOS/nixpkgs/pull/93457** , though AFAIK it doesn't change how narrowly AppArmor is used.

---

nixinator **Nov 2020**

Keep an eye on **https://spectrum-os.org/** , which is currently in development, no release as of yet, but this might be exactly what your looking for 🙂

8573 **Nov 2020**

> nixinator:
>
> Keep an eye on **https://spectrum-os.org/**

Qubes-esque + usability + Nix (+ EU funding) sounds interesting, even if subscribing to their announcements was a bit annoying with the email account I wanted to use (details are in a [details] block because they're… rather off-topic).

► (In case anyone else wants to subscribe to Spectrum OS's announcements with Gmail on Android, here's what I did.)

Atemu **Dec 2020**

I was able to subscribe by simply entering my email **here** and sending an empty reply (same subject) to the email it sent me.

New & Unread Topics

Topic	Replies	Views	Activity
/nix/store/<hash>-grub-2.06/sbin/grub-install: error: 'boot' doesn't look like an EFI partition	**1**	296	**Sep 2024**

Topic	Replies	Views	Activity
Packaging Protonmail’s Import-Export app (QT5)	0	266	Feb 2024
☑ Pipewire host not found and is inactive	2	1.1k	Mar 2024
☑ How to make home-manager available on the CLI in NixOS?	2	2.7k	Mar 2024
Access NixOS SOPS Secret via Home Manager	12	1.9k	Dec 2024

Want to read more? Browse other topics in or view latest topics.