



wayclip

Design & Architectural Report

Name	Ilya Rasputin
Organization	Wayclip
Date	October 13, 2026

1 Abstract

Wayclip is an ecosystem of tools providing Linux users – and in the future Windows users – a new way to create, manage and upload clips. Wayclip was initially developed specifically for Wayland, however, is now growing to more distributions and compositors.

Wayclip solves the core problem of the lack of open-source clipping utilities for gamers on Linux that provide an easy, dedicated and an interactive way to make and share clips. Wayclip achieves this by providing a custom daemon in addition to a range of tools. These tools include a CLI, the Wayclip Application and most notably the Wayclip Platform, enabling the sharing of clips online and engagement between communities. The Wayclip daemon is analogous to other popular applications, such as NVIDIA ShadowPlay, AMD ReLive, and more, but works by interacting directly with your compositor to pipe the output from PipeWire into GStreamer to be processed by your graphics card.

What makes Wayclip different from other projects is our dedication to the open-source community, by providing a lightweight and ready-to-use solution whilst protecting your privacy. This project has a small group of developers, which allows us to perfect and express our ideas more clearly, making Wayclip a better project. Everyday, we are working towards making Wayclip a more organised, more efficient and a more user-friendly environment.

This paper will focus on providing a more detailed outline on the features mentioned, as well as our goal and current development status.

Contents

1 Abstract i

2 The Problem Space 1

2.1 Core problem of *Wayland* 1

2.2 Existing Solutions 1

2.2.1 GPU Screen Recorder 1

2.2.2 Vice 2

2.2.3 OBS Studio 2

2.2.4 wf-recorder 2

2.2.5 Generalisation 2

3 Objectives & Constraints 3

3.1 Core Objectives 3

4 Daemon Architecture 4

5 Wayclip Ecosystem 5

6 Conclusion 6

References 7

2 The Problem Space

2.1 Core problem of *Wayland*

Wayland is the language protocol that applications can use to talk to a display server in order to make themselves visible and get input from the user. A *Wayland* server is called a *compositor*. [1]

To understand the problem, let's look at an older server, a direct competitor of *Wayland* – *X11*. On *X11*, any app can grab any other window's pixels, read keystrokes and inject input events. This is the way everything "just worked" on *X11* [2]. This introduced many security flaws and vulnerabilities, where keyloggers and screen scrapers can run without any friction.

Wayland, on the other hand, flips this model – where each client only knows about its own surfaces. No protocol is able to read another clients' pixels or grab its input [2]. This security change, brings safety to users, but stops all the previous methods from working. Instead, *Wayland* introduces a new portal which asks user for consent before performing any action. The so called *XDG Desktop Portal* brings a framework for securely accessing resources from outside an application sandbox, providing a range of common features to applications, e.g. screencast, screenshots and more [3].

Using this *XDG Desktop Portal*, which in turn will call the `org.freedesktop.portal.ScreenCast` API and ask the user for consent. Afterwards, the portal hands back a *PipeWire* stream that the app reads. *PipeWire* is a project which aims to improve the handling of audio and video under Linux by providing low-latency, graph-based processing engine on top of audio and video devices [4]. *PipeWire* provides the main glue which holds many applications and systems alive, like *Wayclip*.

In short, a convoluted chain of command has to be passed through to screencast, which makes it a pain for developers to introduce new applications. *Wayclip*, instead, leverages this open space to introduce an open-source solution that will bring more support and drive in this community.

2.2 Existing Solutions

Without denial, there already exist (partial) solutions to the problem of screencapture on *Wayland*. Let's break down all the alternatives and compare them to *Wayclip*.

2.2.1 GPU Screen Recorder

GPU Screen Recorder project is a screen recorder which has minimal impact on performance by recording using the GPU only [5]. From internal research, this project works by capturing a DMA-Buf source using *PipeWire*, which is then scanned directly out over a custom socket into their client. The client is built entirely in C with custom GL shaders and `libav` API's. This project gives the option for users to store the encoded frames directly on the RAM which would amount to about 150MB when recording at 60fps, FHD for 2 minutes. These are great results, and the project really does seem to have everything under control, offering a range of codecs and hardware support.

Wayclip, takes a different approach to capturing the screen. Instead of working with low-level binaries and protocols, we use `gstreamer` and `pipewire-libs` bindings directly inside of Rust. This allows us to manage the pipeline with the newest tools in a safe manner. *GStreamer* is a library for constructing graphs of media-handling components. Applications

can take advantage of advances in codec and filter technology transparently. Developers can add new codecs and filters by writing a simple plugin with a clean, generic interface [6].

In addition to the simpler and more manageable interface via *GStreamer*, we also offer built-in support for key-binds, Discord RPC, previews and more.

2.2.2 Vice

Vice, or more commonly known as *Vice Clipper*, is another popular project providing instant-replay game clipping for Linux; press one key to save the last 20 seconds of gameplay [7]. Internally, *Vice* is using the `gpu-screen-recorder` engine, with a python UI and API wrapper. The wrapper, arguably, has some impressive features: video editor, discord link sharing, and automatic game detection. The hotkey detection is also off-loaded to third-party packages via `evdev`. Currently, *Vice* is seen as the leading project in the community due to its large reach and community contributions.

Overall, the project has great features, the main centerpiece being the UI/UX. However, Wayclip offers a more full-inclusive range of features including a CLI, GUI, Website and more. In addition to that, as mentioned previously, the recording in Wayclip is not done via an external tool such as `gpu-screen-recorder`, but is rather built in house using *GStreamer* and the `ashpd` libraries using **Rust**.

2.2.3 OBS Studio

OBS Studio is a free and open source software for video recording and live streaming [8]. *OBS Studio* has gained a good reputation over time as one of the best softwares in this industry. *OBS Studio* comes packaged with a "Replay Buffer" feature, allowing to store the encoded frames in a ring buffer in RAM. *OBS Studio* has a very mature and grown ecosystem of C/C++ tools and packages which support the screencapture across various codecs, backends and hardware. The UI is built using `Qt5/Qt6`, but the project overall has a large footprint, if the goal is solely to use the "Replay Buffer" feature. This project also handles fullscreen game capture with ease using their `obs-vkcapture` hook layer directly, before it reaches Wayland.

This is also a great project, which certainly posed as inspiration for Wayclip, but faces the same problem of lacking the specifics that make a clipping software manageable and easy to use.

2.2.4 wf-recorder

Lastly, *wf-recorder* is a utility program for screen recording of `wlroots`-based compositors (more specifically, those that support `wlr-screencopy-v1` and `xdg-output`) [9]. This project does not have a so-called "Ring Buffer", hence is not able to record in the background and capture the last N seconds. Instead, this software continuously captures the screen and writes it to a file like a linear screen recorder. This project has no UI or an ecosystem, and only supports the "wlroots" environment, making its sole use in this situation – a tool to learn about screen capture.

2.2.5 Generalisation

In general, Wayclip stands out from its competitors by finding the middle ground between bloated desktop softwares (*OBS Studio*) and a bare-bones recording solution (*GPU Screen Recorder*). There, it can provide the performance comparable to the standalone recorders via the use of *GStreamer* pipelines and Rust's memory safety, in addition to the great ecosystem of clip capturing, editing and sharing.

3 Objectives & Constraints

3.1 Core Objectives

Wayclip consists of a small team of developers, so the goals are quite simple and straightforward, but still require persistent effort to achieve.

- **Recording:** Wayclip aims to have a so-called circular "Ring Buffer", able to capture and retain the last N seconds of footage. Then, via the IPC, or a hotkey, a snapshot of the Ring Buffer is taken, and data written to a file. The recording itself should be efficient (meaning low CPU, RAM & VRAM usage), widely supported across codecs and hardware and stable enough to record long gaming session across multiple games. Lastly, the recording should not depend on any third-party packages which already provide a ready tool, meaning Wayclip is not "just another wrapper" and has its own pipeline. *P.S. This does **not** mean that Wayclip will not use any external libraries such as GStreamer, FFmpeg, PipeWire and more...*
- **Ease Of Use:** Wayclip should provide an easy way to install, setup and use the tools provided. This can be achieved by providing official packages across many distributions and consistent support to the community. To ensure that Wayclip is as easy to use, we are planning to implement an intuitive GUI application, alongside a website which prompts user to interact with the clips and explore the ecosystem even further.
- **Universal Support:** Wayclip aims to be available and supported on a range of Linux distributions, and will work identically. Wayclip should also support a variety of hardware such as NVIDIA (NVENC/GL), AMD/Intel (VA-API) and even software based capturing.
- **Rich and Robust Ecosystem:** Wayclip aims to provide an experience of a social media, with a personal home page, comments, friend system and more via the upcoming website. This should be integrated directly with the CLI/GUI, allowing for seamless uploads, management of clips and user interaction.
- **Open-Source:** The most important goal of Wayclip is to provide a base for future projects, and the community to learn and base off. By making the daemon, core library, CLI, GUI (upcoming) and Website/API (Possibly) open-source, we engage directly with the community, allowing for better user feedback, software improvements and increased trust. All the source code is available on our GitHub, under the MIT license.

4 Daemon Architecture

5 Wayclip Ecosystem

6 Conclusion

Bibliography

- [1] freedesktop.org. Wayland architecture. <https://wayland.freedesktop.org/>. Accessed: 2026-10-13.
- [2] Botmonster Tech. Wayland screen sharing: XDG portal, PipeWire fix. <https://botmonster.com/self-hosting/wayland-screen-sharing-fix-video-calls-linux/>. Accessed: 2026-10-13.
- [3] ArchWiki. XDG Desktop Portal. https://wiki.archlinux.org/title/XDG_Desktop_Portal. Accessed: 2026-10-13.
- [4] PipeWire Project. Pipewire. <https://pipewire.org/>. Accessed: 2026-10-13.
- [5] dec05eba. GPU Screen Recorder. <https://git.dec05eba.com/gpu-screen-recorder/about/>. Accessed: 2026-10-13.
- [6] GStreamer Project. GStreamer. <https://gstreamer.freedesktop.org/>. Accessed: 2026-10-13.
- [7] Vice Project. Vice: Open-Source Game Clipping Tool for Linux. <https://github.com/eklonofficial/Vice>. Accessed: 2026-10-13.
- [8] OBS Project. Open Broadcaster Software (OBS Studio). <https://obsproject.com/>. Accessed: 2026-10-13.
- [9] ammen99. wf-recorder: Screen Recorder for wlroots-based Compositors. <https://github.com/ammen99/wf-recorder>. Accessed: 2026-10-13.