Secure Image Editing

Nicolas Schaievitch and Kishan Sreenivasan

Eigen Games 2025

Secure Image Editing

Motivation

Current solutions in image editing and processing, such as AI-based filters, fall under two categories:

- You can edit images locally, which is computationally expensive, and requires you to have the software downloaded to your device
- You use a cloud solution, and give up your privacy by giving your image to some third party.

This same issue happens for many other tasks. Virtually all current SaaS solutions in the market require you to give up the privacy of your data to use their software.

We think this is unacceptable

Our solution

We are leveraging FHE (Full Homomorphic Encryption) and the Othentic stack to allow users to safely and trustlessly submit images for processing. The user encrypts the image, sends the encrypted task, and can decrypt the result without any other party ever seeing the image.

Infrastructure

- We use Zama's TFHE-rs library for the homomorphic encryption
- We use Rust for the client and for the server logic, with a JS layer to handle the validation/execution services
- We use Othentic to develop and deploy our AVS
- We implement a convolution-based image sharpening algorithm

Limitations

- FHE is really slow. We currently only process small grayscale images. As this technology improves, our same logic can scale accordingly.
- We currently use dummy keys for the same reason. Our code can be easily adapted to run with real keys, which a more powerful server could do.

Future improvements

- This same workflow could be used for more complicated tasks, such as object recognition
- Currently, our client is a CLI. In the future, we'd like to compile this to WASM to run on a frontend
- It's expensive to have every validator re-run the FHE computation. Ideally, some other proof of computation could be provided, such as a ZK proof or some probabilistic method.