Mopro:
Make Client-side Proving on Mobile Easy

Vivian Jeng, Moven Tsai

Developer & Grantee, PSE

Section 1

Goal

Facts

- **Mobile users are growing in number compared to desktop users.**
  - Accessibility
  - Portability

- **Native mobile devices are more powerful than browsers.**
  - Performance optimization
  - Access to device hardware (camera, GPS, biometrics,...)
  - Offline functionality
  - More security and privacy
  - Push notifications

- **Mobile development lacks infrastructure for ZK applications.**

Goal

- **Improve developer experience**
  - FFI
  - CLI
  - Templates/Documentation
- **Improve performance**
  - Native binaries
  - GPU resources
- **Build a mobile development ecosystem as complete as the web app ecosystem**
  - Packages
  - Community

Section 2

Roadmap

Roadmap

Adapters

circom

- Arkworks
- witnesscalc
- Rapidsnark
- Tachyon
- Halo2
- Folding
- MPC
  Multi-Party Computation
- FHE
  Fully Homomorphic Encryption

mopro

Platforms

iOS

- Swift
- Objective-C

Android

- kotlin
- Java

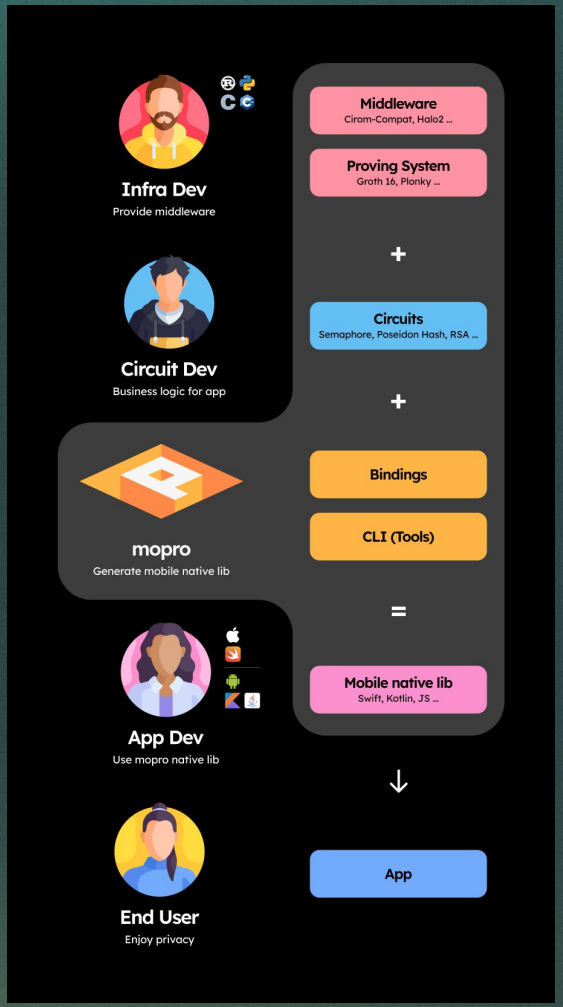
Other

- Web
- Games
- MacOS
- Windows
- Linux

Present · Work in progress

Structure

- **Backend**
  - Proving systems
  - GPU acceleration
- **Middleware**
  - FFI
  - CLI
- **Frontend**
  - SDKs for different protocols
    - Semaphore
    - Zk-email
    - Anon Aadhaar
    - TLSNotary
    - ...



Section 3

Usage

Command-line Interface

Initialize

```
$ mopro init
```

Build

```
$ mopro build
```

Create

```
$ mopro create
```

Section 4

Benchmark

Benchmark for Circom

	snarkjs (on MacOS)	Native app (on iPhone)
Semaphore	902ms	257ms (~3.5x)
Anon Aadhaar	26s	11s (~2.3x)
Keccak256	8406ms	1247ms (~6.7x)
SHA256	2537ms	572ms (~4.4x)
ZK-Email	160s	Out of memory

- **Details:** https://zkmopro.org/docs/performance

Section 5

GPU Acceleration on Client-Side Proving

GPU on client-side is a completely different story than server-side GPU

Client-side vs Server-side

	Apple A17	NVIDIA A100
Compute Power	~2.1 TFLOPS	~19.5 TFLOPS (~9x)
Memory Capacity	8 GB	80 GB (10x)
Scalability	**Standalone**	**Clusterable**
Shading Language	Metal	CUDA

Target on MSM for Proving Time Acceleration

Instance size	Arkworks 0.4 (CPU, M3)	Zprize 2023 Yrrid and Snarkify (CPU*, M2 Pro)	Zprize 2023 Tal and Koh's (GPU**, M2 MAX)
2^16	138 ms	134 ms	832 ms
2^18	491 ms	331 ms	1351 ms
2^20	1922 ms	1230 ms	3188 ms

* WASM runs entirely on CPU
** WebGPU runs most of the tasks

Benchmark on BLS12-377 curve

Details: zprize.io/blog/announcing-the-2023-zprize-winners

Takeaway

- **Under exploration**
  - Better elliptic curve library for Metal
  - Memory-efficient {storage format, algorithms}
  - Exhaust all computing resources
- **CPU and GPU synchronization is the key to accelerate**
- **Client-side GPU is a different story. Let’s explore more!**

https://linktr.ee/zkmopro

Thank you!

Vivian Jeng

Software Engineer, PSE
vivianjeng@pse.dev
tg: @vivianjeng

Moven Tsai

Grantee, PSE
moven0831@gmail.com
tg: @moven0831