Smoo.th

# Even Faster secp256r1 for Passkeys and SGX

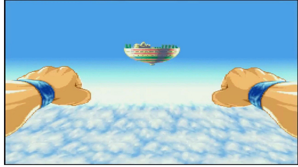
Renaud Dubois

07/04/2024

# Summary

- Introduction
- A story of secp256r1
- Use cases
    - Passkeys
    - SGX
- Cryptographic Gibberish (optimizations and tradeoffs)
- Call to L2s for DSM (double scalar multiplication) EIP/RIP

# Introduction



## 0xDECE65 (codename)

Cryptography team-building tools that make self-sovereignty delightful for users and developers.
Our mantra: there is no Web2 or Web3, just Web.

Some of our current focus:

- Support of the secp256r1 (P256) curve on-chain to enable the use of physical enclaves/keychain
- Support of the ed25559 curve on-chain for the similar reasons
- Support of the FIDO2 framework on-chain, including WebAuthn and the Passkeys
- Development of multi/threshold signature schemes such as FROST, MUSIG2

## The Team

1 security/hardware, 1 smarcontract dev, 1 sales, 1 **cryptographer**

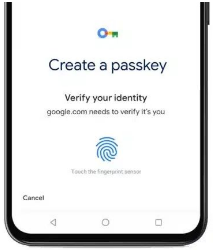
- 17 years in French industry of Defense
- 2 years in Ethereum ecosystem
- Watch our feet, not our mouth
    a. Finalist team at EthGlobal NY 2023
    b. Rewarded by retroPGF for fastest secp256r1 and abstract wallet ( invisible wallet on passkeys)
    c. Granted by Ethereum Foundation for ECC works (half funded)

# A Story of Secp256r1

## What is it

- secp256r1/P256 is an elliptic curve. Elliptic curve cryptography (ECC) is what enables signing and key exchange in modern communications.
- ECDSA over P256 secures our daily lives with our TLS exchanges, some Passports; Intel SGX,  SSH and passkeys (TouchID/FaceID).
- Specified in NIST SP186-5, 20 years of existence
- Non native to ethereum (secp256k1 is native), must be emulated with EVM instructions
- Natural candidate for Account Abstraction

# Passkeys + Account Abstraction : the invisible wallet



Create a passkey

Verify your identity

google.com needs to verify it's you

Touch the fingerprint sensor

Cancel

Account Abstraction + Secp256r1

## What it does:

One click onboarding, invisible for user, vanishing check out churn/difficulties :

- by using the touch ID, creates a  keypair stored in Apple/Android stored in keychain/enclave (device bound)
- the related key is written in the UserOp of ERC4337 as the signer
- front translates transaction to sign to the WebAuthn/Passkeys webbrowser

# SGX + ZK/optimist Rollup : “2FA” settlement

https://ethresear.ch/t/2fa-zk-rollups-using-sgx/14462
(live on scroll)

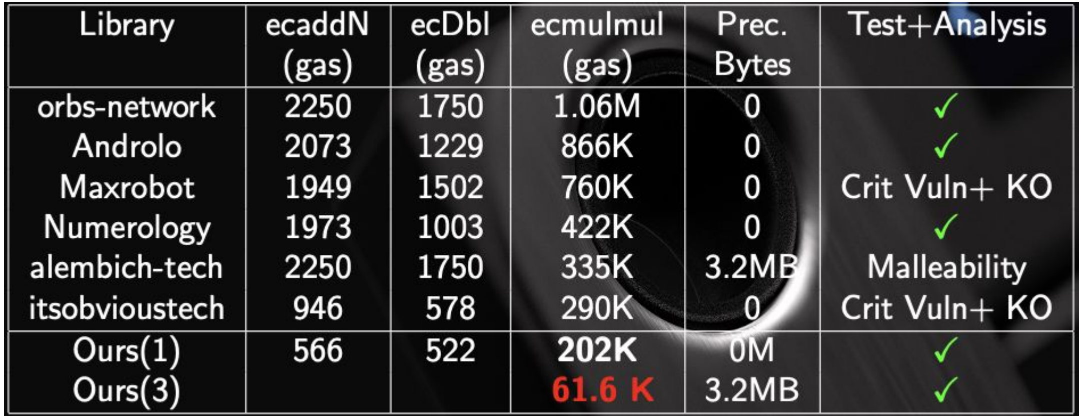
two state transition proofs to advance the on-chain zk-rollup state root:

1. cryptographic proof: a SNARK/STARK
2. 2FA:  additional SGX proof

Intel SGX recently switched from BN curves to secp256r1.

- ZK mechanisms are novel (but the way), and there will be implementation failures (Nova).
- SGX is a way to add an extra layer to buy time for the patch (ZK > SGX)
- Using our precomputation version could save 90% of gas per settlement (scroll, Taiko)

# Past work



| Library | ecaddN (gas) | ecDbl (gas) | ecmulmul (gas) | Prec. Bytes | Test+Analysis |
|---|---|---|---|---|---|
| orbs-network | 2250 | 1750 | 1.06M | 0 | ✓ |
| Androlo | 2073 | 1229 | 866K | 0 | ✓ |
| Maxrobot | 1949 | 1502 | 760K | 0 | Crit Vuln+ KO |
| Numerology | 1973 | 1003 | 422K | 0 | ✓ |
| alembich-tech | 2250 | 1750 | 335K | 3.2MB | Malleability |
| itsobvioustech | 946 | 578 | 290K | 0 | Crit Vuln+ KO |
| Ours(1) | 566 | 522 | **202K** | 0M | ✓ |
| Ours(3) | | | **61.6 K** | 3.2MB | ✓ |

https://eprint.iacr.org/2023/939

## What we did:

Improved by a large factor prior implementations

- Improved by a large factor prior implementations
- Pick optimal formulas according to EVM costs
- Optimize stack
- Use many memory optimization
- Propose precomputations to speed further (complicated for B2C)

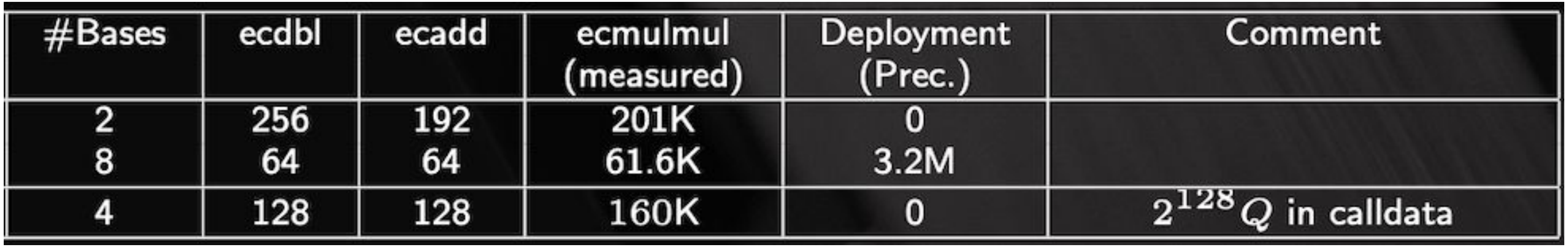
# New Results

## Generic

Having a generic precompile would prevent switching or adding endless precompiles

- A seamless tradeoff, only requires to extend the public key by 512 bits (one extra point)
- What we implemented is DSM (double scalar mul), MSM-EIP is a dream, maybe DSM is the way to begin
- Application of this single precompile :
  - All schnorr based wallet without the “hacky-mul” trick (Ambire)
  - Starkcurve, secp256k1, palla, vesta, babyjujub (circom), ed25519, bn254-G1, eecc2024

## Even faster



| #Bases | ecdbl | ecadd | ecmulmul (measured) | Deployment (Prec.) | Comment |
|---|---|---|---|---|---|
| 2 | 256 | 192 | 201K | 0 | |
| 8 | 64 | 64 | 61.6K | 3.2M | |
| 4 | 128 | 128 | 160K | 0 | $2^{128}Q$ in calldata |

- A seamless tradeoff, only requires to extend the public key by 512 bits (one extra point)

# One EIP to bind them all

Currently many proposals

- secp256r1 (RIP/EIP7212)
- ed25519 (EIP665), ed25519 > secp256r1
  - not NIST, faster, schnorr (MPC/ZK friendly)
  - farcaster, SGX, IBC
- BN254 (EIP1962)
- BLS12381-G1 (EIP 2537)
- BLS12377 (EIP2539)
- Palla/Vesta
- EECC2024

p=0x1000000000000000000000000400010004000000000000000000000000001;

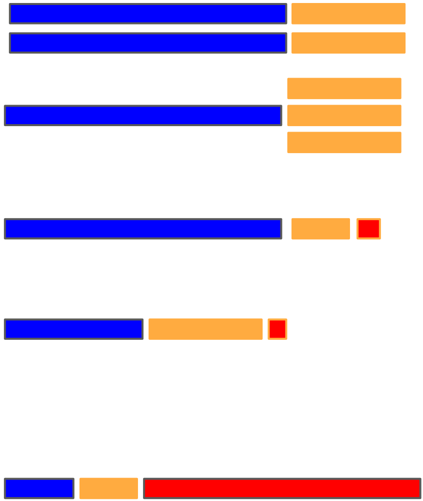
a=0x10000000000000000000020005b002000000000000000000000001 ;

b=0x100000000000000002000000005b000000000200000000000000001;

# Optimizing DSM implementation

4 bases is nearly optimal for 256 bits scalar without precomputations

| Points | Prec ecAdd | EcAdd | EcD bl | Total DSM | Comment |
|---|---|---|---|---|---|
| 1 | | 256 | 512 | 1024 | Double’n Add |
| 2 | 1 | 192 | 256 | 448+1 | Standard Strauss-Shamir DSM |
| 2 | 14 | 60 | 256 | 326+14 | Strauss+Windowing (best RIP-API compliant) |
| 4 | 11 | 120 | 128 | 248+11 | DSM + Precomputed $2^{128}$ P and Q |
| 6 | 57 | 84 | 85 | 169+57 | Cut scalars by 3 is evil |
| 8 | 256 | 64 | 64 | 128+256 | |



(1 point = 64 bytes = 1K gas call data, 12.8K in  contract)

# Cryptographic Optimizations

ECC is easy, you have points, you can add them (P+Q), or double them (P+P=2P) using dedicated formulas.

x 2 =

+

=

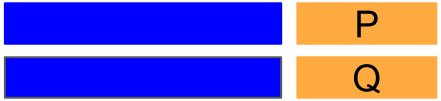
(harder to resurrect Goku, cause there is more points that atoms in the universe)

# The principle

- Double always,  Add when ‘1’

  5P  (b101) : P->2P->4P+P

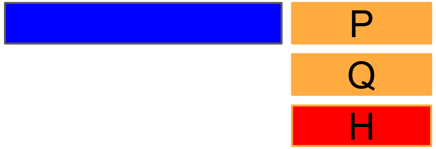
  7Q : (b111): Q->2Q+Q=3Q, 3Q+Q=7Q



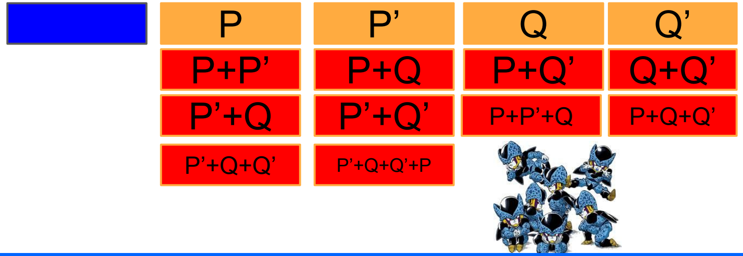
- Strauss Shamir : mutualize doubling, compute H=P+Q

  101 , (P+Q), 2(P+Q)+Q, 2(2P+3Q))+P+Q

  111



- Higher dimension : choose $2^{(n/2)}$P and $2^{(n/2)}$Q as extra points

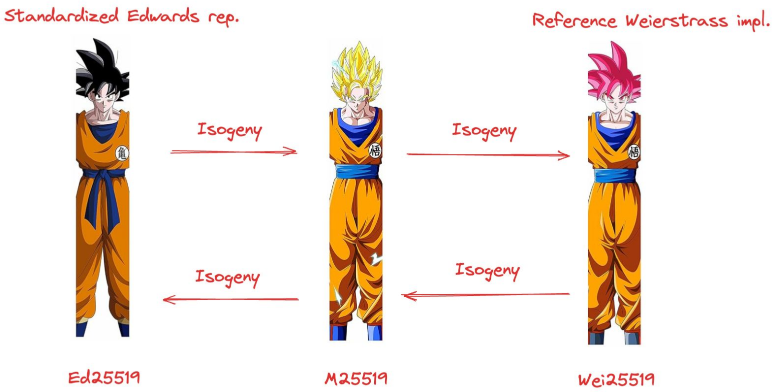


# Isogenies

An isogeny between two elliptic curves and  is a morphism of curves that sends the origin of E1 to the origin of E2.

It is possible, using those morphisms to convert point from edwards representation to Weierstrass and vice versa:

- babyjujub
- ed25519



DSM can be used to implement ed25519 as well with the same code.

# Conclusion

Call for DSM-RIP

- efficient generic ECC progressive precompile
- add generic ECC capacities to the EVM
- release code April, 22th (input to audit)