# Families of SNARK-friendly 2-chains of elliptic curves

**Youssef El Housni**[1] Aurore Guillevic[2]

[1]ConsenSys / Ecole Polytechnique / Inria Saclay

[2]Université de Lorraine / Inria Nancy / Aarhus University

EUROCRYPT, June 2022

# Overview

1. Preliminaries
   - Zero-knowledge proof
   - ZK-SNARK
   - Recursive ZK-SNARKs

2. Contributions: Families of 2-chains
   - Constructions
   - Implementations

# Overview

1. Preliminaries
   - Zero-knowledge proof
   - ZK-SNARK
   - Recursive ZK-SNARKs

2. Contributions: Families of 2-chains
   - Constructions
   - Implementations

"I have a *complete*, *sound* and *zero-knowledge* proof that a statement is true".

### Complete

True statement $\implies$ honest prover convinces honest verifier

### Sound

False statement $\implies$ cheating prover cannot convince honest verifier (except with small proba)

### Zero-knowledge

True statement $\implies$ verifier learns nothing more than statement is true

# ZK-SNARK

## Zero-Knowledge Succinct Non-interactive ARgument of Knowledge

"I have a *complete*, *computationally sound*, *zero-knowledge*, *succinct*, *non-interactive* proof that a statement is true and that I know a related secret".

### Succinct

Honestly-generated proof is very "short" and "easy" to verify.

### Non-interactive

No interaction between the prover and verifier for proof generation and verification.

### ARgument of Knowledge

Honest verifier is convinced that a comptutationally bounded prover knows a secret information.

# ZK-SNARK

## Preprocessing ZK-SNARK of NP language

Let $F$ be a public NP program, $x$ and $z$ be public inputs, and $w$ be a private input such that $z := F(x, w)$.
A ZK-SNARK consists of algorithms $S, P, V$ s.t. for a security parameter $\lambda$:

$$\text{Setup:} \quad (pk, vk) \quad \leftarrow \quad S(F, 1^\lambda)$$

# ZK-SNARK

## Preprocessing ZK-SNARK of NP language

Let $F$ be a public NP program, $x$ and $z$ be public inputs, and $w$ be a private input such that $z := F(x, w)$.
A ZK-SNARK consists of algorithms $S, P, V$ s.t. for a security parameter $\lambda$:

| | | | |
|---|---|---|---|
| Setup: | $(pk, vk)$ | $\leftarrow$ | $S(F, 1^\lambda)$ |
| Prove: | $\pi$ | $\leftarrow$ | $P(x, z, w, pk)$ |

# ZK-SNARK

## Preprocessing ZK-SNARK of NP language

Let $F$ be a public $\texttt{NP}$ program, $x$ and $z$ be public inputs, and $w$ be a private input such that $z := F(x, w)$.
A ZK-SNARK consists of algorithms $S, P, V$ s.t. for a security parameter $\lambda$:

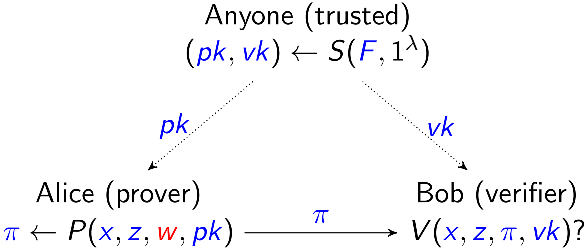
| | | | |
|---|---|---|---|
| Setup: | $(pk, vk)$ | $\leftarrow$ | $S(F, 1^\lambda)$ |
| Prove: | $\pi$ | $\leftarrow$ | $P(x, z, w, pk)$ |
| Verify: | $\texttt{false/true}$ | $\leftarrow$ | $V(x, z, \pi, vk)$ |

# ZK-SNARK

## Preprocessing ZK-SNARK of NP language

Let $F$ be a public NP program, $x$ and $z$ be public inputs, and $w$ be a private input such that $z \coloneqq F(x, w)$.
A ZK-SNARK consists of algorithms $S, P, V$ s.t. for a security parameter $\lambda$:

| | | | |
|---|---|---|---|
| Setup: | $(pk, vk)$ | $\leftarrow$ | $S(F, 1^\lambda)$ |
| Prove: | $\pi$ | $\leftarrow$ | $P(x, z, w, pk)$ |
| Verify: | `false/true` | $\leftarrow$ | $V(x, z, \pi, vk)$ |



- $E\colon y^2 = x^3 + ax + b$ elliptic curve defined over $\mathbb{F}_q$, $q$ a prime power.
- $r$ prime divisor of $\#E(\mathbb{F}_q) = q + 1 - t$, $t$ Frobenius trace.
- $k$ embedding degree, smallest integer $k \in \mathbb{N}^*$ s.t. $r \mid q^k - 1$.
- a bilinear pairing

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$$

  - $\mathbb{G}_1 \subset E(\mathbb{F}_q)$ a group of order $r$
  - $\mathbb{G}_2 \subset E(\mathbb{F}_{q^k})$ a group of order $r$.
  - $\mathbb{G}_T \subset \mathbb{F}^*_{q^k}$ group of $r$-th roots of unity.

**Example: Groth16 [Gro16]**

Given $z := F(x, w)$ where $(x, z, w) = (x_0, \ldots, x_i, z_{i+1}, \cdots, z_\ell, w_{\ell+1}, \cdots, w_n)$

**Example: Groth16 [Gro16]**

Given $z := F(x, w)$ where $(x, z, w) = (x_0, \ldots, x_i, z_{i+1}, \cdots, z_\ell, w_{\ell+1}, \cdots, w_n)$

- $(pk, vk) \leftarrow S(F, 1^\lambda)$ where

$$pk \in \mathbb{G}_1^{2n+\ell+3} \times \mathbb{G}_2^{\ell+2}, \quad vk \in \mathbb{G}_1^{\ell+1} \times \mathbb{G}_2^2 \times \mathbb{G}_T$$

**Example: Groth16 [Gro16]**

Given $z := F(x, w)$ where $(x, z, w) = (x_0, \ldots, x_i, z_{i+1}, \cdots, z_\ell, w_{\ell+1}, \cdots, w_n)$

- $(pk, vk) \leftarrow S(F, 1^\lambda)$ where

$$pk \in \mathbb{G}_1^{2n+\ell+3} \times \mathbb{G}_2^{\ell+2}, \quad vk \in \mathbb{G}_1^{\ell+1} \times \mathbb{G}_2^2 \times \mathbb{G}_T$$

- $\pi \leftarrow P(x, z, w, pk)$ where

$$\pi = (A, B, C) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \qquad (O_\lambda(1))$$

**Example: Groth16 [Gro16]**

Given $z := F(x, w)$ where $(x, z, w) = (x_0, \ldots, x_i, z_{i+1}, \cdots, z_\ell, w_{\ell+1}, \cdots, w_n)$

- $(pk, vk) \leftarrow S(F, 1^\lambda)$ where

$$pk \in \mathbb{G}_1^{2n+\ell+3} \times \mathbb{G}_2^{\ell+2}, \quad vk \in \mathbb{G}_1^{\ell+1} \times \mathbb{G}_2^2 \times \mathbb{G}_T$$

- $\pi \leftarrow P(x, z, w, pk)$ where

$$\pi = (A, B, C) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \qquad (O_\lambda(1))$$

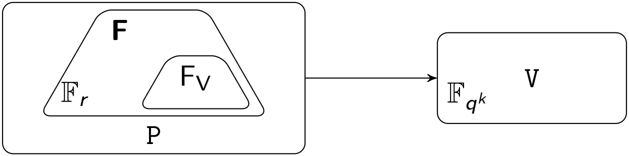
- `false/true` $\leftarrow V(x, z, \pi, vk)$ where $V$ is

$$e(A, B) \stackrel{?}{=} vk_1 \cdot e(vk_2', vk_3) \cdot e(C, vk_4) \qquad (O_\lambda(\ell)) \qquad (*)$$

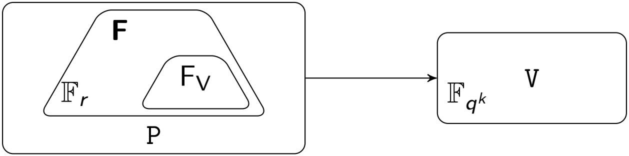
and $vk_2' = \sum_{i=0}^{\ell} [x_i] vk_2$.

# Recursive ZK-SNARKs

## An arithmetic mismatch




- **F** any program is expressed in $\mathbb{F}_r$
- P proving is performed over $\mathbb{G}_1$ (and $\mathbb{G}_2$) (of order $r$)
- V verification (eq. *) is done in $\mathbb{F}^*_{q^k}$
- $F_V$ program of V is natively expressed in $\mathbb{F}^*_{q^k}$ not $\mathbb{F}_r$




**F** any program is expressed in $\mathbb{F}_r$

P proving is performed over $\mathbb{G}_1$ (and $\mathbb{G}_2$) (of order $r$)

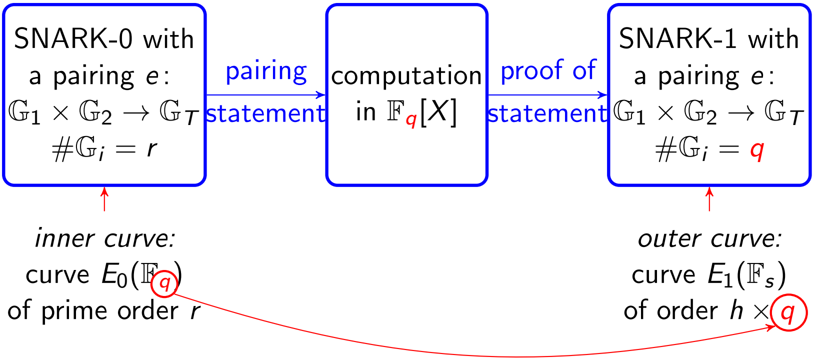
V verification (eq. *) is done in $\mathbb{F}^*_{q^k}$

$F_V$ program of V is natively expressed in $\mathbb{F}^*_{q^k}$ not $\mathbb{F}_r$

- $1^{st}$ attempt: choose a curve for which $q = r$ (impossible)
- $2^{nd}$ attempt: simulate $\mathbb{F}_q$ operations via $\mathbb{F}_r$ operations ($\times \log q$ blowup)
- $3^{rd}$ attempt: use a cycle/chain of pairing-friendly elliptic curves [CFH+15, BCTV14, BCG+20]

# Recursive ZK-SNARKs

## A proof of a proof




Given $q$, search for a pairing-friendly curve
$E_1$ of order $h \cdot q$ over a field $\mathbb{F}_s$

# Overview

# Choice of elliptic curves
## ZK-curves

- SNARK
  - $E/\mathbb{F}_q$ — BN, BLS12, BW12?, KSS16? ... [FST10]
    - pairing-friendly
    - $r-1$ highly 2-adic (efficient FFT)
- Recursive SNARK (2-cycle)
  - $E_1/\mathbb{F}_{q_1}$ and $E_2/\mathbb{F}_{q_2}$ — MNT4/MNT6 [FST10, Sec.5], ? [CCW19]
    - both pairing-friendly
    - $r_2 = q_1$ and $r_1 = q_2$
    - $r_{\{1,2\}} - 1$ highly 2-adic (efficient FFT)
    - $q_{\{1,2\}} - 1$ highly 2-adic (efficient FFT)
- Recursive SNARK (2-chain)
  - $E_1/\mathbb{F}_{q_1}$ — BLS12 ($seed \equiv 1 \bmod 3 \cdot 2^{large}$) [BCG+20], ?
    - pairing-friendly
    - $r_1 - 1$ highly 2-adic
    - $q_1 - 1$ highly 2-adic
  - $E_2/\mathbb{F}_{q_2}$ — Cocks–Pinch, Brezing–Weng
    - pairing-friendly
    - $r_2 = q_1$

# Choice of elliptic curves

## Curve $E_2/\mathbb{F}_{q_2}$

- $q$ is a prime or a prime power
- $t$ is relatively prime to $q$
- ~~$r$ is prime~~
- ~~$r$ divides $q+1-t$~~
- ~~$r$ divides $q^k-1$ (smallest $k \in \mathbb{N}^*$)~~
- $4q - t^2 = Dy^2$ (for $D < 10^{12}$) and some integer $y$

} $r$ is a **fixed** chosen prime that divides $q+1-t$ and $q^k-1$ (smallest $k \in \mathbb{N}^*$)

---

**Algorithm 1:** Cocks–Pinch method

1 Fix $k$ and $D$ and choose a prime $r$ s.t. $k | r-1$ and $\left(\frac{-D}{r}\right) = 1$;
2 Compute $t = 1 + x^{(r-1)/k}$ for $x$ a generator of $(\mathbb{Z}/r\mathbb{Z})^\times$;
3 Compute $y = (t-2)/\sqrt{-D} \bmod r$;
4 Lift $t$ and $y$ in $\mathbb{Z}$;
5 Compute $q = (t^2 + Dy^2)/4$ (in $\mathbb{Q}$);
6 back to 1 if $q$ is not a prime integer;

---

# 2-chains

## Limitations and improvements

- $\rho = \log_2 q / \log_2 r \approx 2$ (because $q = f(t^2, y^2)$ and $t, y \xleftarrow{\$} \bmod r$).
- The curve parameters $(q, r, t)$ are not expressed as polynomials.

---

**Algorithm 2:** Brezing–Weng method

1 Fix $k$ and $D$ and choose an irreducible polynomial $r(x) \in \mathbb{Z}[x]$ with positive leading coefficient [1] s.t. $\sqrt{-D}$ and the primitive $k$-th root of unity $\zeta_k$ are in $K = \mathbb{Q}[x]/r(x)$;

2 Choose $t(x) \in \mathbb{Q}[x]$ be a polynomial representing $\zeta_k + 1$ in $K$;

3 Set $y(x) \in \mathbb{Q}[x]$ be a polynomial mapping to $(\zeta_k - 1)/\sqrt{-D}$ in $K$;

4 Compute $q(x) = (t^2(x) + Dy^2(x))/4$ in $\mathbb{Q}[x]$;

---

- $\rho = 2 \max(\deg t(x), \deg y(x)) / \deg r(x) < 2$
- $r(x), q(x), t(x)$ but $q(x)$ (never) irreducible!
- lift $t = t(x_0) + h_t r$ and $y = y(x_0) + h_y r$

[1] conditions to satisfy Bunyakovsky conjecture which states that such a polynomial produces infinitely many primes for infinitely many integers.

# Inner curves

## SNARK-0

**Groth16 SNARK**

- 128-bit security
- pairing-friendly
- efficient $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ and pairing
- $p - 1 \equiv r - 1 \equiv 0 \mod 2^L$ for large input $L \in \mathbb{N}^*$ (FFTs)

$\rightarrow$ BLS ($k = 12$) family of roughly 384 bits with seed $x \equiv 1 \mod 3 \cdot 2^L$

**Universal-KZG SNARK**

- 128-bit security
- pairing-friendly
- efficient $\mathbb{G}_1$, ~~$\mathbb{G}_2$, $\mathbb{G}_T$~~ and pairing
- $p - 1 \equiv r - 1 \equiv 0 \mod 2^L$ for large $L \in \mathbb{N}^*$ (FFTs)

$\rightarrow$ BLS ($k = 24$) family of roughly 320 bits with seed $x \equiv 1 \mod 3 \cdot 2^L$

# Outer curves

## SNARK-1

**Groth16 SNARK**

- 128-bit security
- pairing-friendly
- efficient $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ and pairing
- $r' = p$ $(r' - 1 \equiv 0 \mod 2^L)$

$\rightarrow$ BW $(k = 6)$ family of roughly 768 bits with $(t \mod x) \mod r \equiv 0$ or 3

**Universal-KZG SNARK**

- 128-bit security
- pairing-friendly
- efficient $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ and pairing
- $r' = p$ $(r' - 1 \equiv 0 \mod 2^L)$

$\rightarrow$ BW $(k = 6)$ family of roughly 704 bits with $(t \mod x) \mod r \equiv 0$ or 3
$\rightarrow$ CP $(k = 8)$ family of roughly 640 bits
$\rightarrow$ CP $(k = 12)$ family of roughly 640 bits

*All $\mathbb{G}_i$ formulae and pairings are given in terms of $x$ and some $h_t, h_y \in \mathbb{N}$.*

We short list few 2-chains of the proposed families that have some additional nice engineering properties

- Groth16: BLS12-377 and BW6-761
- Universal-KZG: BLS24-315 and BW6-633 (or BW6-672)

Table: Cost of $\mathtt{S}$, $\mathtt{P}$ and $\mathtt{V}$ algorithms for Groth16 and Universal-KZG. $n$ =number of multiplication gates, $a$ =number of addition gates and $\ell$ =number of public inputs. $\mathtt{M}_{\mathbb{G}}$ =multiplication in $\mathbb{G}$ and $\mathtt{P}$=pairing.

| | $\mathtt{S}$ | $\mathtt{P}$ | $\mathtt{V}$ |
|---|---|---|---|
| Groth16 | $3n\ \mathtt{M}_{\mathbb{G}_1},\ n\ \mathtt{M}_{\mathbb{G}_2}$ | $(4n-\ell)\ \mathtt{M}_{\mathbb{G}_1},\ n\ \mathtt{M}_{\mathbb{G}_2}$ | $3\ \mathtt{P},\ \ell\ \mathtt{M}_{\mathbb{G}_1}$ |
| Universal-KZG | $d_{\geq n+a}\ \mathtt{M}_{\mathbb{G}_1},\ 1\ \mathtt{M}_{\mathbb{G}_2}$ | $9(n+a)\ \mathtt{M}_{\mathbb{G}_1}$ | $2\ \mathtt{P},\ 18\ \mathtt{M}_{\mathbb{G}_1}$ |

# Implementation and benchmark

https://github.com/ConsenSys/gnark (Go)

$F_V$: program that checks V (eq. *) ($\ell = 1$, ~~$n = 80000$~~ $n = 19378$)

Table: Groth16 (ms)

| | S | P | V |
|---|---|---|---|
| BLS12-377 | 387 | 34 | 1 |
| BLS24-315 | 501 | 54 | 4 |
| BW6-761 | 1226 | 114 | 9 |
| BW6-633 | 710 | 69 | 6 |
| BW6-672 | 840 | 74 | 7 |

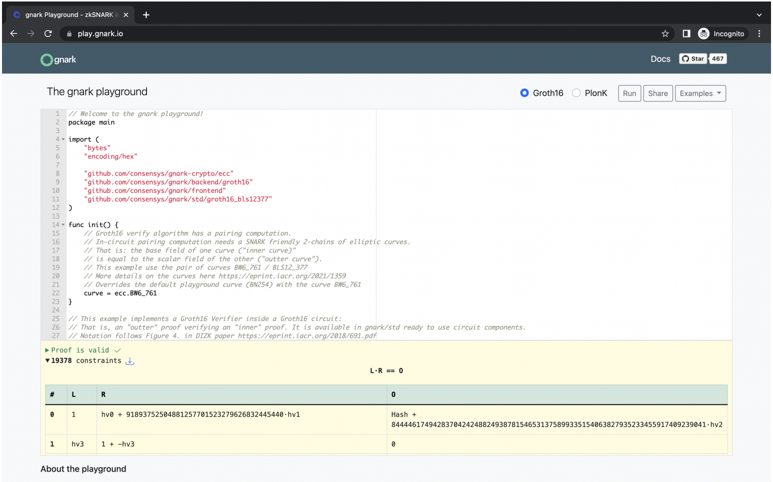
Table: Universal-KZG (ms)

| | S | P | V |
|---|---|---|---|
| BLS12-377 | 87 | 215 | 4 |
| BLS24-315 | 76 | 173 | 1 |
| BW6-761 | 294 | 634 | 9 |
| BW6-633 | 170 | 428 | 6 |
| BW6-672 | 190 | 459 | 7 |

# Play with gnark!

Write SNARK programs at https://play.gnark.io/

*Example*: Proof of Groth16 `V` program (eq. *)



# Conclusion

paper ePrint 2021/1359

implementations github/ConsenSys/gnark-crypto (Go)
gitlab/inria/snark-2-chains (SageMath/MAGMA)

follow-up work Survey of elliptic curves for proof systems ePrint 2022/586

follow-up work Pairings in Rank-1 Constraint System (to be submitted)

follow-up work Co-factor clearing and subgroup membership on pairing-friendly elliptic curves ePrint 2022/352 (AFRICACRYPT 2022)

THANK YOU!

# References I


- Sean Bowe, Alessandro Chiesa, Matthew Green, Ian Miers, Pratyush Mishra, and Howard Wu.
  Zexe: Enabling decentralized private computation.
  In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1059–1076, Los Alamitos, CA, USA, may 2020. IEEE Computer Society.

- Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza.
  Scalable zero knowledge via cycles of elliptic curves.
  In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 276–294. Springer, Heidelberg, August 2014.

- Alessandro Chiesa, Lynn Chua, and Matthew Weidner.
  On cycles of pairing-friendly elliptic curves.
  *SIAM Journal on Applied Algebra and Geometry*, 3(2):175–192, 2019.

# References II


Craig Costello, Cédric Fournet, Jon Howell, Markulf Kohlweiss, Benjamin Kreuter, Michael Naehrig, Bryan Parno, and Samee Zahur.
Geppetto: Versatile verifiable computation.
In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 253–270. IEEE Computer Society, 2015.
ePrint 2014/976.

David Freeman, Michael Scott, and Edlyn Teske.
A taxonomy of pairing-friendly elliptic curves.
*Journal of Cryptology*, 23(2):224–280, April 2010.

# References III


Jens Groth.
On the size of pairing-based non-interactive arguments.
In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 305–326. Springer, Heidelberg, May 2016.