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

JC2, April 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 $\texttt{NP}$ 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 λ:

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

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 λ:

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

Let F be a public $\mathtt{NP}$ 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 λ:

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

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 λ:

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

$$
\begin{tikzcd}
 & \begin{array}{c}\text{Anyone (trusted)}\\ (pk, vk) \leftarrow S(F, 1^\lambda)\end{array} \arrow[dl, dotted, "pk"'] \arrow[dr, dotted, "vk"] & \\
\begin{array}{c}\text{Alice (prover)}\\ \pi \leftarrow P(x, z, w, pk)\end{array} \arrow[rr, "\pi"] & & \begin{array}{c}\text{Bob (verifier)}\\ V(x, z, \pi, vk)?\end{array}
\end{tikzcd}
$$

ZK-SNARK

Pairing-based preprocessing ZK-SNARK of NP language

- $E\colon y^2 = x^3 + ax + b$ 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]

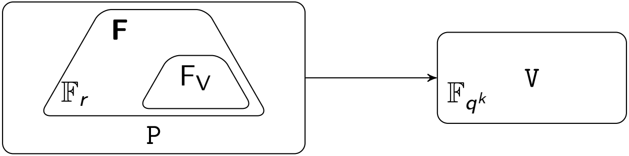
Given $z := F(x, w)$ where $(x, z, w) = (x_0, \ldots, x_i, z_{i+1}, \cdots, z_\ell, w_{\ell+1}, \cdots, w_n)$

- $(pk, vk) \leftarrow S(F, 1^\lambda)$ 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
$$\pi = (A, B, C) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \qquad (O_\lambda(1))$$

- $\texttt{false/true} \leftarrow V(x, z, \pi, vk)$ where V is
$$e(A, B) \stackrel{?}{=} vk_1 \cdot e(vk_2', vk_3) \cdot e(C, vk_4) \qquad (O_\lambda(\ell)) \qquad (*)$$

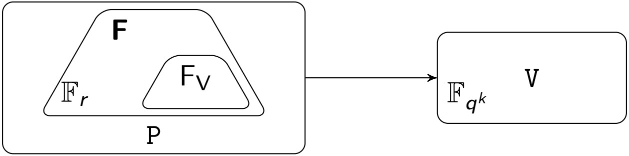
and $vk_2' = \sum_{i=0}^{\ell} [x_i] vk_2$.



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

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 $\mathbb{F}_r$

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

Recursive ZK-SNARKs

A proof of a proof

SNARK-0 with a pairing e: $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ $\#\mathbb{G}_i = r$

pairing statement

computation in $\mathbb{F}_q[X]$

proof of statement

SNARK-1 with a pairing e: $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ $\#\mathbb{G}_i = q$

inner curve:
curve $E_0(\mathbb{F}_q)$
of prime order r

outer curve:
curve $E_1(\mathbb{F}_s)$
of order $h \times q$

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

Overview

1. Preliminaries
   - Zero-knowledge proof
   - ZK-SNARK
   - Recursive ZK-SNARKs
2. Contributions: Families of 2-chains
   - Constructions
   - Implementations

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 SNARK

- 128-bit security
- pairing-friendly
- efficient $\mathbb{G}_1$, $\cancel{\mathbb{G}_2}$, $\cancel{\mathbb{G}_T}$ 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 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 $h_t, h_y \in \mathbb{N}$.

Implementation and benchmark

Short-list of curves

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

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

Table: Cost of $\mathtt{S}$, $\mathtt{P}$ and $\mathtt{V}$ algorithms for Groth16 and Universal. n =number of multiplication gates, a =number of addition gates and ℓ =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	$d_{\geq n+a}\ \mathtt{M}_{\mathbb{G}_1}$, $1\ \mathtt{M}_{\mathbb{G}_2}$	$9(n+a)\ \mathtt{M}_{\mathbb{G}_1}$	$2\ \mathtt{P}$, $18\ \mathtt{M}_{\mathbb{G}_1}$

Implementation and benchmark

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

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

Table: Groth16 (ms)

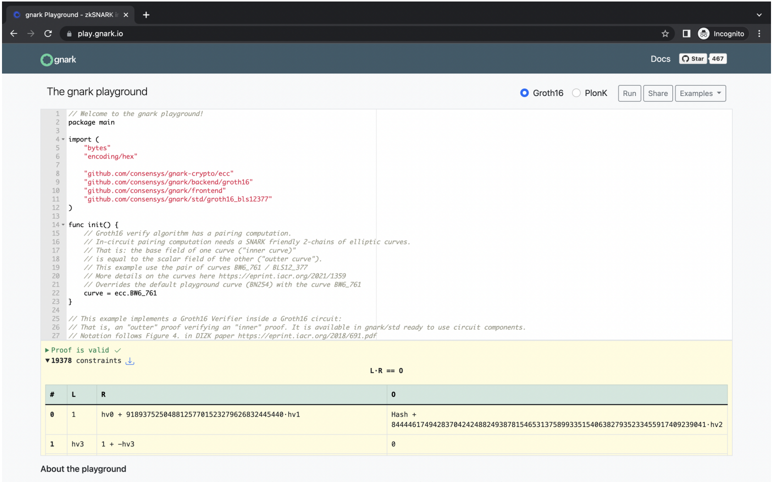
	S	P	V
BLS12-377	387	34	1
BLS24-315	501	54	4
BW6-761	1226	114	9
BW6-633	710	69	6
BW6-672	840	74	7

Table: Universal (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 (accepted at EUROCRYPT 2022)

implementations github/ConsenSys/gnark-crypto (Go)

gitlab/inria/snark-2-chains (SageMath/MAGMA)

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

ongoing work Survey of elliptic curves for SNARKs (soon on ePrint)

Pairings in Rank-1 Constraint System (implemented + paper WIP)

THANK YOU!

Take ~~away~~ your train

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.

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.

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.