A survey of elliptic curves for proof systems

Diego Aranha, Youssef El Housni, *Aurore Guillevic*
With many slides from Youssef El Housni

Aarhus University and Inria Nancy

June 24, 2022

https://members.loria.fr/AGuillevic/files/talks/22-06-Nancy.pdf

Our work

Diego F. Aranha, Youssef El Housni, and Aurore Guillevic.
A survey of elliptic curves for proof systems.
Cryptology ePrint Archive, Paper 2022/586, 2022.
https://eprint.iacr.org/2022/586.

Youssef El Housni and Aurore Guillevic.
Families of SNARK-friendly 2-chains of elliptic curves.
In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022*, volume 13276 of *LNCS*, pages 367–396. Springer, 2022.
ePrint 2021/1359.

Youssef El Housni and Aurore Guillevic.
Optimized and secure pairing-friendly elliptic curves suitable for one layer proof composition.
In Stephan Krenn, Haya Shulman, and Serge Vaudenay, editors, *CANS 20*, volume 12579 of *LNCS*, pages 259–279. Springer, Heidelberg, December 2020.

Outline

Preliminaries on proof systems
- Zero-knowledge proof (ZKP)
- ZK-SNARK

Pairings

Curves for proof systems
- proof composition
- SNARK curves

Pairing-friendly curves
- 2-chains of pairing-friendly elliptic curves

Implementations

Outline

Zero-Knowledge Proofs

Alice

I know the solution to
this complex equation

Bob

No idea what the solution is
but Alice must know it

“Prove it” ←

Challenge ←

Response →

Zero-Knowledge for public keys: Sigma protocol

Alice | **Bob**

I know x such that $g^x = y$

Zero-Knowledge for public keys: Sigma protocol

Alice **Bob**

I know x such that $g^x = y$

$r \xleftarrow{\text{random}} \mathbb{Z}_p$ $\xrightarrow{A = g^r}$

Zero-Knowledge for public keys: Sigma protocol

Alice | **Bob**

I know x such that $g^x = y$

$r \xleftarrow{\text{random}} \mathbb{Z}_p$ $\xrightarrow{A = g^r}$

$\xleftarrow{c}$ $c \xleftarrow{\text{random}} \mathbb{Z}_p$

Zero-Knowledge for public keys: Sigma protocol

Alice		**Bob**
I know x such that $g^x = y$		
$r \xleftarrow{\text{random}} \mathbb{Z}_p$	$\xrightarrow{A = g^r}$	
	$\xleftarrow{c}$	$c \xleftarrow{\text{random}} \mathbb{Z}_p$
$s = r + c \cdot x$	$\xrightarrow{s}$	

Zero-Knowledge for public keys: Sigma protocol

Alice		**Bob**
I know x such that $g^x = y$		
$r \xleftarrow{\text{random}} \mathbb{Z}_p$	$\xrightarrow{A = g^r}$	
	$\xleftarrow{c}$	$c \xleftarrow{\text{random}} \mathbb{Z}_p$
$s = r + c \cdot x$	$\xrightarrow{s}$	$g^s \stackrel{?}{=} A \cdot y^c$
		with $A \cdot y^c = g^r \cdot g^{x \cdot c}$
		then $g^r \cdot g^{x \cdot c} = g^{r + x \cdot c}$

Zero-Knowledge for public keys: Sigma protocol

Alice		**Bob**
I know x such that $g^x = y$		
$r \xleftarrow{\text{random}} \mathbb{Z}_p$	$\xrightarrow{A = g^r}$	
	$\xleftarrow{c}$	$c \xleftarrow{\text{random}} \mathbb{Z}_p$
$s = r + c \cdot x$	$\xrightarrow{s}$	$g^s \stackrel{?}{=} A \cdot y^c$
		with $A \cdot y^c = g^r \cdot g^{x \cdot c}$
		then $g^r \cdot g^{x \cdot c} = g^{r + x \cdot c}$

$x \in \mathbb{Z}_p$, but $A, g, y \in \mathbf{G}$ a group of order p, e.g. $E(\mathbb{F}_q)$, $\#E(\mathbb{F}_q) = p$

Non-Interactive Zero-Knowledge (NIZK) Sigma protocol

Alice

Bob

I know x such that $g^x = y$

$$r \xleftarrow{\text{random}} \mathbb{Z}_p$$
$$A = g^r$$
$$c = H(A, y)$$
$$s = r + c \cdot x$$

$$\xrightarrow{\pi = (A, c, s)}$$

$$g^s \stackrel{?}{=} A \cdot y^c$$
$$c \stackrel{?}{=} H(A, y)$$

Non-Interactive Zero-Knowledge (NIZK) Sigma protocol

Alice

Bob

I know x such that $g^x = y$

$$r \xleftarrow{\text{random}} \mathbb{Z}_p$$
$$A = g^r$$
$$c = H(A, y)$$
$$s = r + c \cdot x$$

$$\xrightarrow{\pi = (A, c, s)}$$

$$g^s \stackrel{?}{=} A \cdot y^c$$
$$c \stackrel{?}{=} H(A, y)$$

$x \in \mathbb{Z}_p$, but $A, g, y \in \mathbf{G}$ a group of order p, e.g. $E(\mathbb{F}_q)$, $\#E(\mathbb{F}_q) = p$

ZKP families

- *specific* statement vs *general* statement
- *interactive* vs *non-interactive* protocol
- *transparent* setup vs *trapdoored* setup vs *no* setup
- *Any* verifier vs *given* verifier
- prover complexity (Alice)
- verifier complexity (Bob)
- communication complexity (size of the proof and the setup)
- security assumptions, cryptographic primitive...
- ...

Blockchains and ZKP

A blockchain is a public peer-to-peer *decentralized*, *transparent, immutable, paying* ledger.

- *Transparent*: everything is visible to everyone
- *Immutable*: nothing can be removed once written
- *Paying*: everyone should pay a fee to use

Transparent $\xrightarrow[\text{Problem}]{}$ confidentiality $\xrightarrow[\text{Solution}]{}$ ZKP

setup, prover?, verifier?

Immutable $\xrightarrow[\text{Problem}]{}$ scalability $\xrightarrow[\text{Solution}]{}$ ZKP

Communication complexity

Paying $\xrightarrow[\text{Problem}]{}$ cost $\xrightarrow[\text{Solution}]{}$ ZKP

Verifier complexity, *prover*?

ZKP literature landmarks

- First ZKP paper [GMR85]
- Non-Interactive ZKP [BFM88]
- Succinct ZKP [K92]
- Succinct Non-Interactive ZKP [M94]
- Succinct NIZK without the PCP Theorem [Groth10]
- "SNARK" terminology and characterization of existence [BCCT11]
- Succinct NIZK without PCP Theorem and Quasi-linear prover time [GGPR13]
- Succinct NIZK without with constant-size proof and constant-time verifier (Groth16)
- First succinct NIZK with universal and updatable setup [Sonic19]
- Active research and implementation on SNARK with universal and updatable setup [PLONK19]
- ...

Zero-knowledge proof

What is a zero-knowledge proof?

"I have a *sound*, *complete* and *zero-knowledge* proof that a statement is true". [GMR85]

Sound

False statement $\implies$ cheating prover cannot convince honest verifier.

Complete

True statement $\implies$ honest prover convinces honest verifier.

Zero-knowledge

True statement $\implies$ verifier learns nothing other than statement is true.

Zero-knowledge proof

ZK-SNARK: Zero-Knowledge Succinct Non-interactive ARgument of Knowledge

"I have a *computationally sound*, *complete*, *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 computationally 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 λ:

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

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:

$$\begin{array}{llll} \text{Setup:} & (pk, vk) & \leftarrow & S(F, 1^{\lambda}) \\ \text{Prove:} & \pi & \leftarrow & P(x, z, w, pk) \end{array}$$

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:	$\texttt{false/true}$	$\leftarrow$	$V(x, z, \pi, vk)$

ZK-SNARK

Preprocessing ZK-SNARK of NP language

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:	$\mathtt{false/true}$	$\leftarrow$	$V(x, z, \pi, vk)$

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

ZK-SNARK

Succinctness: An honestly-generated proof is very "short" and "easy" to verify.

Definition [BCTV14b]

A succinct proof π has size $O_\lambda(1)$ and can be verified in time $O_\lambda(|F| + |x| + |z|)$, where $O_\lambda(.)$ is some polynomial in the security parameter λ.

ZK-SNARKs in a nutshell

main ideas:

1. Reduce a "general statement" satisfiability to a polynomial equation satisfiability.
2. Use Schwartz-Zippel lemma to succinctly verify the polynomial equation with high probability.
3. Use homomorphic hiding cryptography to blindly verify the polynomial equation.
4. Use Fiat-Shamir transform to make the protocol non-interactive.

To know more about zk-SNARK, see Youssef slides at Aarhus seminar, May 11, 2022.

Outline

What is a pairing?

$(\mathbf{G}_1, +), (\mathbf{G}_2, +), (\mathbf{G}_T, \cdot)$ three cyclic groups of large prime order ℓ
Pairing: map $e : \mathbf{G}_1 \times \mathbf{G}_2 \to \mathbf{G}_T$

1. bilinear: $e(P_1 + P_2, Q) = e(P_1, Q) \cdot e(P_2, Q)$, $e(P, Q_1 + Q_2) = e(P, Q_1) \cdot e(P, Q_2)$
2. non-degenerate: $e(G_1, G_2) \neq 1$ for $\langle G_1 \rangle = \mathbf{G}_1$, $\langle G_2 \rangle = \mathbf{G}_2$
3. efficiently computable.

Most often used in practice:

$$e([a]P, [b]Q) = e([b]P, [a]Q) = e(P, Q)^{ab} \ .$$

$\rightsquigarrow$ Many applications in asymmetric cryptography.

Pairings in cryptography: 1993 and 2001

1993

Menezes–Okamoto–Vanstone attack on supersingular curves

2001

- Joux' tri-partite key exchange
- Boneh Frankin Identity based encryption
- Boneh Lynn Shacham short signature

Pairing setting: elliptic curves

$$E/\mathbb{F}_p: \; y^2 = x^3 + ax + b, \; a, b \in \mathbb{F}_p, \; p \geq 5$$

- proposed in 1985 by Koblitz, Miller
- $E(\mathbb{F}_p)$ has an efficient group law (chord an tangent rule) $\rightarrow$ $\mathbf{G}_1$
- $\#E(\mathbb{F}_p) = p + 1 - t$, trace t: $|t| \leq 2\sqrt{p}$
- efficient group order computation (*point counting*)

Pairing setting: elliptic curves

$$E/\mathbb{F}_p: \ y^2 = x^3 + ax + b, \ a, b \in \mathbb{F}_p, \ p \geq 5$$

- proposed in 1985 by Koblitz, Miller
- $E(\mathbb{F}_p)$ has an efficient group law (chord an tangent rule) $\to$ $\mathbf{G}_1$
- $\#E(\mathbb{F}_p) = p + 1 - t$, trace t: $|t| \leq 2\sqrt{p}$
- efficient group order computation (*point counting*)
- large subgroup of prime order ℓ s.t. $\ell \mid p + 1 - t$ and ℓ coprime to p
- $E(\mathbb{F}_p)[\ell] = \{P \in E(\mathbb{F}_p) \colon [\ell]P = \mathcal{O}\}$ has order ℓ
- $E[\ell] \simeq \mathbb{Z}/\ell\mathbb{Z} \times \mathbb{Z}/\ell\mathbb{Z}$ (for crypto)
- only generic attacks against DLP on well-chosen genus 1 and genus 2 curves
- optimal parameter sizes

Tate Pairing and modified Tate pairing

$\ell \mid p^n - 1,\ E[\ell] \subset E(\mathbb{F}_{p^n})$

Tate Pairing: For cryptography,

- $\mathbf{G}_1 = E(\mathbb{F}_p)[\ell] = \{P \in E(\mathbb{F}_p), [\ell]P = \mathcal{O}\}$
- embedding degree $n > 1$ w.r.t. ℓ: smallest[1] integer n s.t. $\ell \mid p^n - 1$
  $\Leftrightarrow E[\ell] \subset E(\mathbb{F}_{p^n})$
- $\mathbf{G}_2 \subset E(\mathbb{F}_{p^n})[\ell]$
- $\mathbf{G}_1 \cap \mathbf{G}_2 = \mathcal{O}$ by construction for practical applications
- $\mathbf{G}_T = \boldsymbol{\mu}_\ell = \{u \in \mathbb{F}_{p^n}^*,\ u^\ell = 1\} \subset \mathbb{F}_{p^n}^*$

When n is small i.e. $1 \leqslant n \leqslant\sim 50$, the curve is *pairing-friendly*.

This is very rare: For a given curve, $\log n \sim \log \ell$ (Balasubramanian–Koblitz).

[1] $n = 1$ is possible too in rare circumstances

Modified Tate pairing

Ensure the pairing is non-degenerate: $\mathbf{G}_1 \cap \mathbf{G}_2 = \mathcal{O}$

$$E[\ell] = \mathbb{Z}/\ell\mathbb{Z} \times \mathbb{Z}\ell\mathbb{Z} = \mathbf{G}_1 \times \mathbf{G}_2$$

Let $P \in \mathbf{G}_1 = E(\mathbb{F}_p)[\ell], Q \in \mathbf{G}_2 \subset E(\mathbb{F}_{p^n})[\ell]$.
Let $f_{\ell,P}$ the function s. t. $\mathrm{Div}(f_{\ell,P}) = \ell(P) - \ell(\mathcal{O})$.
$f_{\ell,P}$ is a function in $\mathbb{F}_{p^n}[x,y]$ with a zero at P of multiplicity ℓ and a pole at $\mathcal{O}$ of mult. ℓ
Modified Tate pairing (in cryptography):

$$\begin{array}{ccccccc} & E(\mathbb{F}_p)[\ell] & & E(\mathbb{F}_{p^n})[\ell] & & \\ & \wr\| & & \cup & & \\ e_{\mathrm{Tate}}: & \mathbf{G}_1 & \times & \mathbf{G}_2 & \to & \boldsymbol{\mu}_\ell \subset \mathbb{F}_{p^n}^* \\ & & (P,Q) & & \mapsto & (f_{\ell,P}(Q))^{\frac{p^n-1}{\ell}} \end{array}$$

Miller Loop

Input: integer s, points P, Q of order s

Output: $m = f_{s,P}(Q)$, where $\mathrm{Div}(f) = s(P) - s(\mathcal{O})$

$m \leftarrow 1$; $S \leftarrow P$;

for *b from the second most significant bit of s to the least* **do**

  $\ell \leftarrow \ell_{S,S}(Q)$; $S \leftarrow [2]S$; `// Double Line`

  $v \leftarrow v_{[2]S}(Q)$; `// Vertical Line`

  $m \leftarrow m^2 \cdot \ell / v$; `// Update 1`

  if $b = 1$ **then**

    $\ell \leftarrow \ell_{S,P}(Q)$; $S \leftarrow S + P$; `// Add Line`

    $v \leftarrow v_{S+Q}(Q)$; `// Vertical Line`

    $m \leftarrow m \cdot \ell / v$; `// Update 2`

return m;

Outline

Proof composition

A proof

Example: Groth16 [Gro16]

Given an instance $\Phi = (a_0, \ldots, a_\ell) \in \mathbb{F}_r^\ell$ of a public NP program F

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

$$vk = (vk_{\alpha,\beta}, \{vk_{\pi_i}\}_{i=0}^{\ell}, vk_\gamma, vk_\delta) \in \mathbf{G}_T \times \mathbf{G}_1^{\ell+1} \times \mathbf{G}_2 \times \mathbf{G}_2$$

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

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

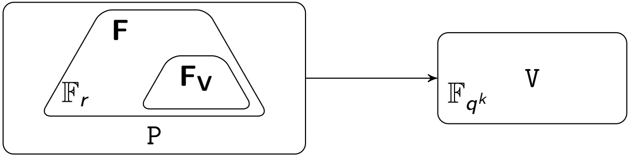
- $0/1 \leftarrow V(\Phi, \pi, vk)$ where V is

$$e(A, B) = vk_{\alpha,\beta} \cdot e(vk_x, vk_\gamma) \cdot e(C, vk_\delta) \qquad (O_\lambda(|\Phi|)) \tag{1}$$

and $vk_x = \sum_{i=0}^{\ell} [a_i] vk_{\pi_i}$ depends only on the instance Φ and $vk_{\alpha,\beta} = e(vk_\alpha, vk_\beta)$ can be computed in the trusted setup for $(vk_\alpha, vk_\beta) \in \mathbf{G}_1 \times \mathbf{G}_2$.

Recursive ZK-SNARKs

An arithmetic mismatch



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

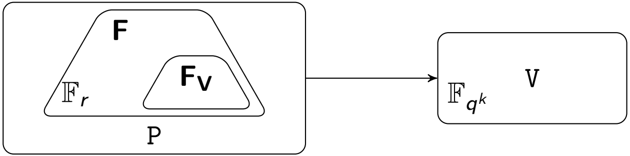
P proving is performed over $\mathbf{G}_1$ (and $\mathbf{G}_2$) (of order r)

V verification (eq. 1) is done in $\mathbb{F}^*_{q^k}$

F_V program of 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 $\mathbf{G}_1$ (and $\mathbf{G}_2$) (of order r)

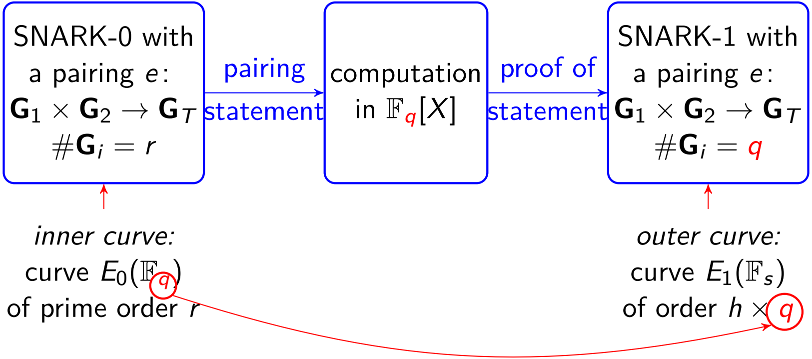
V verification (eq. 1) is done in $\mathbb{F}^*_{q^k}$

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, BCTV14a, 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 $\mathbb{F}_s$

Proof composition

cycles and chains of pairing-friendly elliptic curves

Definition

An m-chain of elliptic curves is a list of distinct curves

$$E_1/\mathbb{F}_{q_1}, \ldots, E_m/\mathbb{F}_{q_m}$$

where $q_1, \ldots, q_m$ are large primes and

$$\#E_2(\mathbb{F}_{q_2}) = q_1, \ldots, \; \#E_i(\mathbb{F}_{q_i}) = q_{i-1}, \ldots, \; \#E_m(\mathbb{F}_{q_m}) = q_{m-1} \; . \tag{2}$$

Definition

An m-cycle of elliptic curves is an m-chain, with

$$\#E_1(\mathbb{F}_{q_1}) = q_m \; . \tag{3}$$

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 algorithm
    - pairing-friendly
    - $r_2 = q_1$

Outline

First ordinary pairing-friendly curves: MNT

Miyaji, Nakabayashi, Takano, $\#E(\mathbb{F}_p) = p(u) + 1 - t(u)$, $r(u) \mid \#E(\mathbb{F}_p)$

k	param	MNT
3	$t(u)$	$-1 \pm 6u$
	$r(u)$	$12u^2 \mp 6u + 1$
	$p(u)$	$12u^2 - 1$
	Dy^2	$12u^2 \pm 12u - 5$
4	$t(u)$	$-u, \ u+1$
	$r(u)$	$u^2 + 2u + 2, \ u^2 + 1$
	$p(u)$	$u^2 + u + 1$
	Dy^2	$3u^2 + 4u + 4$
6	$t(u)$	$1 \pm 2u$
	$r(u)$	$4u^2 \mp 2u + 1$
	$p(u)$	$4u^2 + 1$
	Dy^2	$12u^2 - 4u + 3$

CODA: $k = 6$, 753 bits, ≈ 137 bits of security, $D = -241873351932854907$, seed $u =$ `0xaa3a58eb20d1fec36e5e772ee6d3ff28c296465f137300399db8a5521e18d33581a262716214583d3b89820dd0c000`

Cycle of curves

statement in a group of prime order p over a field $\mathbb{F}_r$

statement in a group of prime order r over a field $\mathbb{F}_p$

elliptic curve $E_0(\mathbb{F}_r)$ of prime order p

elliptic curve $E_1(\mathbb{F}_p)$ of prime order r

MNT-4 and MNT-6 curves form a cycle

$$\begin{array}{llll} k = 4, \text{ MNT-4 parameters} & t_4 = -v, & r_4 = v^2 + 1, & p_4 = v^2 + v + 1 \\ k = 6, \text{ MNT-6 parameters} & t_6 = 1 - 2u, & r_6 = 4u^2 + 2u + 1, & p_6 = 4u^2 + 1 \end{array}$$

$$\begin{array}{c} r_4 = p_6 \\ \text{and} \\ p_4 = r_6 \end{array} \iff \begin{array}{c} v = 2u \\ \text{and} \\ r_4,\ r_6 \text{ are primes} \end{array}$$

Unique known cycle of pairing-friendly curves.
Impossibility results:

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.

Very popular pairing-friendly curves: Barreto-Naehrig (BN)

$$E_{BN} : \; y^2 = x^3 + b, \; p \equiv 1 \bmod 3, \; D = -3 \text{ (ordinary)}$$

$$\begin{aligned} p &= 36x^4 + 36x^3 + 24x^2 + 6x + 1 \\ t &= 6x^2 + 1 \\ \ell &= p + 1 - t = 36x^4 + 36x^3 + 18x^2 + 6x + 1 \\ t^2 - 4p &= -3(6x^2 + 4x + 1)^2 \;\rightarrow\; \text{no CM method needed} \end{aligned}$$

Comes from the Aurifeuillean factorization of Φ_{12} :

$\Phi_{12}(6x^2) = \ell(x)\ell(-x)$

Security level	$\log_2 \ell$	finite field	n	$\log_2 p$	$\deg P, \; p = P(u)$	ρ
102	256	3072	12	256	4	1
123	384	4608	12	384	4	1
132	448	5376	12	448	4	1

BLS12

Barreto, Lynn, Scott method.
Becomes more and more popular, replacing BN curves

$$E_{BLS}: \ y^2 = x^3 + b, \ p \equiv 1 \bmod 3, \ D = -3 \text{ (ordinary)}$$

$$\begin{aligned}
p &= (u-1)^2/3(u^4 - u^2 + 1) + u \\
t &= u + 1 \\
r &= (u^4 - u^2 + 1) = \Phi_{12}(u) \\
p + 1 - t &= (u-1)^2/3(u^4 - u^2 + 1) \\
t^2 - 4p &= -3y(u)^2 \ \rightarrow \ \text{no CM method needed}
\end{aligned}$$

BLS12-381 with seed `-0xd201000000010000`

The Cocks–Pinch method

Three equations:

$$\begin{aligned} \ell &\mid p+1-t && (4) \\ \ell &\mid \Phi_n(p) && (5) \\ t^2-4p &= -Dy^2 && (6) \end{aligned}$$

From (4), $p \equiv t-1 \bmod \ell$

From (5) and (4), $\ell \mid \Phi_n(t-1) \iff t-1 = \zeta_n \bmod \ell$

where ζ_n is a primitive n-th root of unity modulo ℓ, ζ_n exists $\iff \ell \equiv 1 \bmod n$.

$$\boldsymbol{t = \zeta_n + 1 \bmod \ell}$$

From (6) and (4), with $p = (t^2 + Dy^2)/4$,

$$p+1-t = \frac{1}{4}\left(t^2 - 4t + 4 + Dy^2\right) = \frac{1}{4}\left((t-2)^2 + Dy^2\right)$$

Because $\ell \mid p+1-t$, assuming ℓ odd,

$$(t-2)^2 + Dy^2 = 0 \bmod \ell \implies \boldsymbol{y = \frac{t-2}{\sqrt{-D}} \bmod \ell}$$

The Cocks–Pinch method

Input: A positive integer n and a positive square-free integer D
Output: $E/\mathbb{F}_q$ with an order-ℓ subgroup and embedding degree n

1 Choose a prime ℓ such that n divides $\ell - 1$ and $-D$ is a square modulo ℓ
2 Compute $t = 1 + x^{(\ell-1)/n}$ for x a generator of $(\mathbb{Z}/\ell\mathbb{Z})^\times$, $t - 1 \equiv \zeta_n \bmod \ell$
3 Compute $y = (t-2)/\sqrt{-D} \bmod \ell$
4 Lift t and y in $\mathbb{Z}$
5 Compute $q = (t^2 + Dy^2)/4$ in $\mathbb{Q}$
6 **if** *q is a prime integer* **then**
7   Use CM method ($D < 10^{20}$) to get the coefficients of $E/\mathbb{F}_q$ with order-ℓ subgroup
8 **else**
9   Go back to 1
10 **return** $E/\mathbb{F}_q$ *with an order-ℓ subgroup and embedding degree n*

The Cocks–Pinch method

Drawback: $\log|t|, \log|y| \approx \log \ell \implies \log p \approx 2 \log \ell$
rho-value:

$$\rho = \frac{\log p}{\log \ell} \approx 2$$

The optimal would be $\rho = 1$ for a prime-order curve, $\ell = p + 1 - t$.

How to compute primitive n-th roots of unity:

Input: prime ℓ, integer $n > 0$, $\ell \equiv 1 \bmod n$
Output: $\zeta_n \bmod \ell$

$z \leftarrow \mathtt{random}(\ell)$
$z \leftarrow z^{(\ell-1)/n}$
while $\Phi_n(z) \neq 0 \bmod \ell$ *(or: $z^d = 1 \bmod \ell$ for some $d \mid n$, $1 \leq d < n$)* **do**
    $z \leftarrow \mathtt{random}(\ell)$
    $z \leftarrow z^{(\ell-1)/n}$
return z

The CM method (Complex Multiplication)

Hard problem to compute the curve coefficients (a, b) given a prime p and a trace t.
The other way: given p and (a, b) in $E/\mathbb{F}_p\colon\ : y^2 = x^3 + ax + b$ and computing the order $\#E(\mathbb{F}_p)$ is done with the SEA algorithm (Schroof–Elkies–Atkin).
The *CM* method computes a j-invariant, given p, t.

1. Compute the discriminant $-D$ as the square-free part in $t^2 - 4p = -Dy^2$
2. If $D \equiv 1, 2 \bmod 4$, $D \leftarrow 4D$
3. Compute a Hilbert Class Polynomial $H_{-D}(X)$ mod p with Sutherland's software `classpoly` at `https://math.mit.edu/~drew/`
4. Compute a root j_0 of $H_{-D}(X)$ mod p
5. Set $E\colon y^2 = x^3 + \frac{3j_0}{1728-j_0}x + \frac{2j_0}{1728-j_0}$

The CM method

For specific (small) values of $-D$, the j-invariants are known:

- $-D = -3$, $j = 0$
- $-D = -4$, $j = 1728$
- $-D = -8$, $j = 8000$
- $-D = -7$, $j = -3375$
- $-D = -11$, $j = -32768$
- $-D = -19$, $j = -884736$
- $-D = -43$, $j = -884736000$
- $-D = -67$, $j = -147197952000$
- $-D = -163$, $j = -262537412640768000$

The Brezing–Weng method: The Cocks–Pinch method with polynomials

Start with $r(x)$ an irreducible polynomial s.t. the number field $K = \mathbb{Q}[x]/(r(x))$ contains ζ_n and $\sqrt{-D}$

Algorithm 1: Idea of Barreto–Lynn–Scott and Brezing–Weng methods

Input: A positive integer n and a positive square-free integer D

Output: Polynomials $p(x), r(x), t(x)$ s.t. $t^2(x) - 4p(x) = -Dy^2(x)$,
$r(x) \mid p(x) + 1 - t(x)$, $r(x) \mid \Phi_n(p(x))$

1 Choose an irreducible polynomial $r(x) \in \mathbb{Z}[x]$ with positive leading coefficient such that $\sqrt{-D}$ and $\zeta_n \in K = \mathbb{Q}[x]/(r(x))$
2 Choose $t(x) \in \mathbb{Q}[x]$ a polynomial representing $\zeta_n + 1$ in K
3 Set $y(x) \in \mathbb{Q}[x]$ a polynomial mapping to $(\zeta_n - 1)/\sqrt{-D}$ in K
4 Compute $p(x) = (t^2(x) + Dy^2(x))/4$ in $\mathbb{Q}[x]$
5 If $p(x)$ does not represent primes go back to 1 or 2
6 **return** $p(x), r(x), t(x)$

The BLS family

If $3 \mid n$, then $\sqrt{-3} \in K = \mathbb{Q}[x]/(\Phi_n(x))$

- $n = 3$: $\zeta_3 = \frac{-1+\sqrt{-3}}{2} \in \mathbb{C}$, $\Phi_3 = x^2 + x + 1$
  For $n \equiv 3 \bmod 6$, $\zeta_3 = x^{n/3} \bmod \Phi_n(x)$
  $\sqrt{-3} = 2x^{n/3} + 1$ and $1/\sqrt{-3} = \sqrt{-3}/3 = (2x^{n/3} + 1)/3$
- $n = 6$: $\zeta_6 = \frac{11+\sqrt{-3}}{2} \in \mathbb{C}$, $\Phi_6 = x^2 - x + 1$
  For $n \equiv 0 \bmod 6$, $\zeta_6 = x^{n/6} \bmod \Phi_n(x)$
  $\sqrt{-3} = 2x^{n/6} - 1$ and $1/\sqrt{-3} = \sqrt{-3}/3 = (2x^{n/6} - 1)/3$

Given n multiple of 3,

1. $r(x) \leftarrow \Phi_n(x)$
2. $t(x) \leftarrow x + 1$
3. $y(x) \leftarrow (x - 1)/\sqrt{-3}$
   - $y(x) = (x - 1)(2x^{n/3} + 1)/3$ if $n \equiv 3 \bmod 6$
   - $y(x) = (x - 1)(2x^{n/6} - 1)/3$ if $n \equiv 0 \bmod 6$
4. $p(x) = (t^2(x) + 3y^2(x))/4$

Finding 2-chains 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 2: Cocks–Pinch method

1 Fix k and D and choose a prime r s.t. $k|r-1$ and $(\frac{-D}{r}) = 1$;
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 3: 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 ζ_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 $\mathbb{Q}[x]$;

- $\rho = 2 \max\left(\deg t(x), \deg y(x)\right) / \deg r(x) < 2$
- $r(x), q(x), t(x)$ but does $\exists\ x_0 \in \mathbb{Z}^*, r(x_0) = r_{\text{fixed}}$ and $q(x_0)$ is prime ?

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

2-chains

Notes

- $\mathbf{G}_2 \subset E(\mathbb{F}_{q^k}) \cong E'[r](\mathbb{F}_{q^{k/d}})$ for a twist E' of degree d.
- When $-D = -3$, there exists a twist E' of degree $d = 6$.
- Associated with a choice of $\xi \in \mathbb{F}_{q^{k/6}}$ s.t. $x^6 - \xi \in \mathbb{F}_{q^{k/6}}[x]$ is irreducible, the equation of E' can be either
  - $y^2 = x^3 + b/\xi$ and we call it a D-twist or
  - $y^2 = x^3 + b \cdot \xi$ and we call it a M-twist.
- For the D-type, $E' \to E : (x, y) \mapsto (\xi^{1/3}x, \xi^{1/2}y)$,
- For the M-type $E' \to E : (x, y) \mapsto (\xi^{2/3}x/\xi, \xi^{1/2}y/\xi)$

2-chains

Suggested construction: combines CP and BW

1. Cocks–Pinch method
   - $k = 6$ and $-D = -3 \implies$ 128-bit security, $\mathbf{G}_2$ coordinates in $\mathbb{F}_q$, GLV multiplication over $\mathbf{G}_1$ and $\mathbf{G}_2$
   - restrict search to $\text{size}(q) \le 768$ bits $\implies$ smallest machine-word size
2. Brezing–Weng method
   - choose $r(x) = q_{\text{BLS }12-377}(x)$
   - $q(x) = (t^2(x) + 3y^2(x))/4$ factors $\implies$ $q(x_0)$ cannot be prime
   - lift $t = r \times h_t + t(x_0)$ and $y = r \times h_y + y(x_0)$ [FK19, GMT20]

2-chains [CANS2020]

The suggested curve: BW6-761

$E : y^2 = x^3 - 1$ over $\mathbb{F}_q$ of 761-bit with seed $x_0 = \mathtt{0x8508c00000000}$ and polynomials:

Our curve, $k = 6$, $D = 3$, $r = q_{\mathrm{BLS\,12-377}}$
$r(x) = (x^6 - 2x^5 + 2x^3 + x + 1)/3 = q_{\mathrm{BLS\,12-377}}(x)$
$t(x) = x^5 - 3x^4 + 3x^3 - x + 3 + h_t r(x)$
$y(x) = (x^5 - 3x^4 + 3x^3 - x + 3)/3 + h_y r(x)$
$q(x) = (t^2 + 3y^2)/4$
$q_{h_t=13,h_y=9}(x) = (103x^{12} - 379x^{11} + 250x^{10} + 691x^9 - 911x^8$ $-79x^7 + 623x^6 - 640x^5 + 274x^4 + 763x^3 + 73x^2 + 254x + 229)/9$

Inner curves [EC2022]

SNARK-0

Groth16 SNARK

- 128-bit security
- pairing-friendly
- efficient $\mathbf{G}_1$, $\mathbf{G}_2$, $\mathbf{G}_T$ 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 $\mathbf{G}_1$, $\cancel{\mathbf{G}_2}$, $\cancel{\mathbf{G}_T}$ 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 [EC2022]

SNARK-1

Groth16 SNARK

- 128-bit security
- pairing-friendly
- efficient $\mathbf{G}_1$, $\mathbf{G}_2$, $\mathbf{G}_T$ 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 $\mathbf{G}_1$, $\mathbf{G}_2$, $\mathbf{G}_T$ 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 $\mathbf{G}_i$ formulae and pairings are given in terms of x and some $h_t, h_y \in \mathbb{N}$.

Outline

Preliminaries on proof systems

Pairings

Curves for proof systems

Pairing-friendly curves

Implementations

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. $\mathtt{M}_{\mathbf{G}}$ =multiplication in $\mathbf{G}$ and $\mathtt{P}$=pairing.

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

Implementation and benchmark

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

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

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

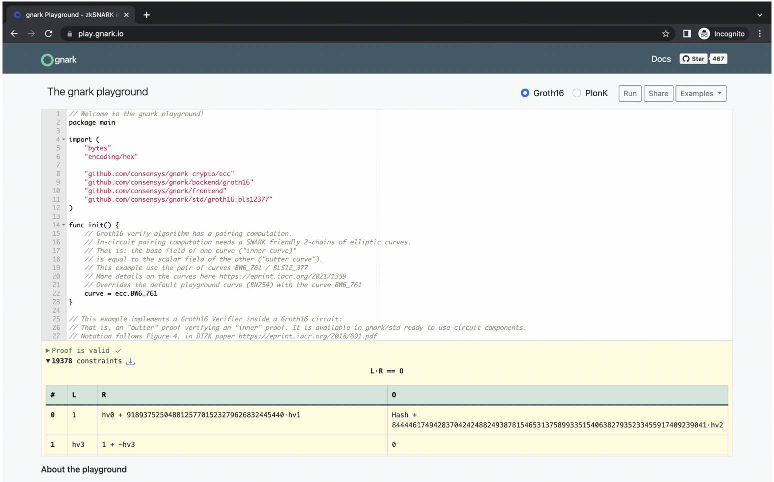
Table: Universal (ms)

	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. 1)



Conclusion

papers	2-chains: ePrint 2021/1359 (EUROCRYPT 2022) Survey of elliptic curves for SNARKs: ePrint 2022/586 (submitted)
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 (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.

Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza.
Succinct non-interactive zero knowledge for a von neumann architecture.
In Kevin Fu and Jaeyeon Jung, editors, *USENIX Security 2014*, pages 781–796. USENIX Association, August 2014.

References II

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.

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.

Georgios Fotiadis and Elisavet Konstantinou.
TNFS resistant families of pairing-friendly elliptic curves.
Theoretical Computer Science, 800:73–89, 31 December 2019.

References III

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

Aurore Guillevic, Simon Masson, and Emmanuel Thomé.
Cocks–Pinch curves of embedding degrees five to eight and optimal ate pairing computation.
Des. Codes Cryptogr., 88:1047–1081, March 2020.

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.