# DiSSECT: Distinguisher of Standard & Simulated Elliptic Curves via Traits

Vladimir Sedlacek[1,2] Vojtech Suchanek[1] Antonin Dufka[1]
Marek Sys [1] Vashek Matyas[1]

[1]CRoCS, Masaryk University, Brno, Czech Republic

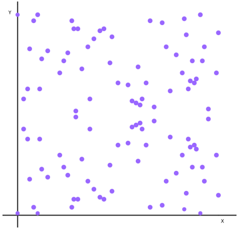
[2]Université de Picardie Jules Verne, Amiens, France

Africacrypt 2022, July 20

# Elliptic curve cryptography (ECC)

$$y^2 = x^3 + ax + b \quad \text{in } \mathbb{F}_p$$

$$k \cdot P := \underbrace{P + \cdots + P}_{k}$$

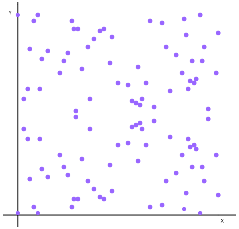


- ECC based on the discrete logarithm problem (ECDLP): Given $P$, $k \cdot P$, find $k$

# Elliptic curve cryptography (ECC)

$$y^2 = x^3 + ax + b \quad \text{in } \mathbb{F}_p$$

$$k \cdot P := \underbrace{P + \cdots + P}_{k}$$

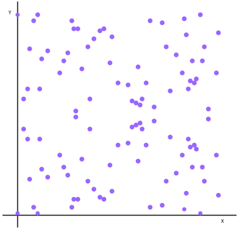


- ECC based on the discrete logarithm problem (ECDLP): Given $P, k \cdot P$, find $k$
- Protocols: ECDH, ECDSA, EdDSA

# Elliptic curve cryptography (ECC)

$$y^2 = x^3 + ax + b \quad \text{in } \mathbb{F}_p$$

$$k \cdot P := \underbrace{P + \cdots + P}_{k}$$

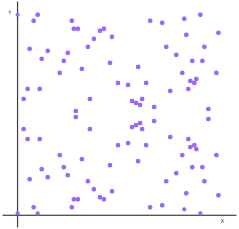


- ECC based on the discrete logarithm problem (ECDLP): Given $P, k \cdot P$, find $k$
- Protocols: ECDH, ECDSA, EdDSA
- In practice: standard curves

# Elliptic curve cryptography (ECC)

$$y^2 = x^3 + ax + b \quad \text{in } \mathbb{F}_p$$

$$k \cdot P := \underbrace{P + \cdots + P}_{k}$$

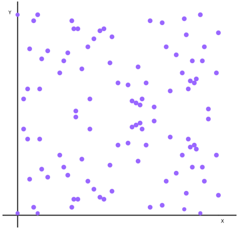


- ECC based on the discrete logarithm problem (ECDLP): Given $P, k \cdot P$, find $k$
- Protocols: ECDH, ECDSA, EdDSA
- In practice: standard curves
  - Who chooses them and how?

# Elliptic curve cryptography (ECC)

$$y^2 = x^3 + ax + b \quad \text{in } \mathbb{F}_p$$

$$k \cdot P := \underbrace{P + \cdots + P}_{k}$$



- ECC based on the discrete logarithm problem (ECDLP): Given $P$, $k \cdot P$, find $k$
- Protocols: ECDH, ECDSA, EdDSA
- In practice: standard curves
  - Who chooses them and how?
  - How to measure their real security?

# ECDLP attacks

- Known attacks
- Unknown attacks
- Backdoors

# Known attacks

- Based on known and published weaknesses

# Known attacks

- Based on known and published weaknesses
  - Pohlig-Hellman

# Known attacks

- Based on known and published weaknesses
  - Pohlig-Hellman
  - MOV and SASS attack

# Known attacks

- Based on known and published weaknesses
  - Pohlig-Hellman
  - MOV and SASS attack
  - Small CM-discriminant

# Known attacks

- Based on known and published weaknesses
  - Pohlig-Hellman
  - MOV and SASS attack
  - Small CM-discriminant
- All of these depend just on $p$ and the group order

# Known attacks

- Based on known and published weaknesses
  - Pohlig-Hellman
  - MOV and SASS attack
  - Small CM-discriminant
- All of these depend just on $p$ and the group order
- Fairly easy to avoid

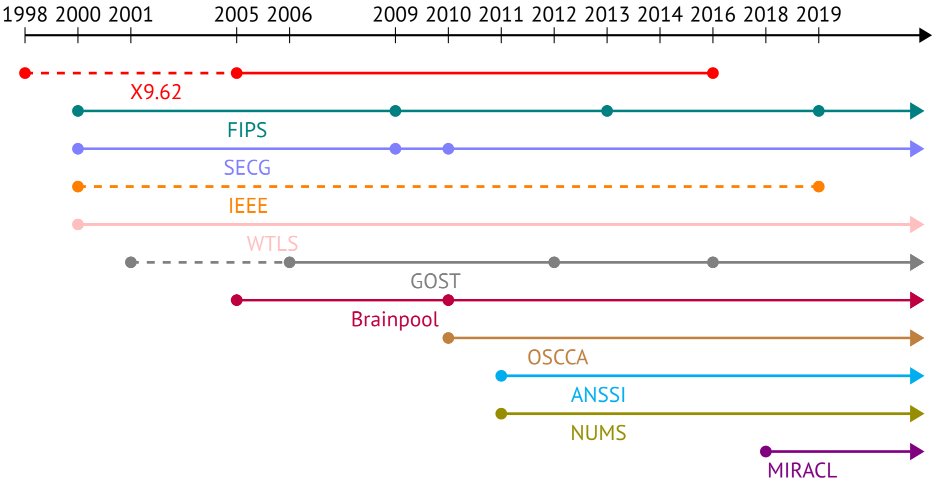
# Known attacks

- Based on known and published weaknesses
  - Pohlig-Hellman
  - MOV and SASS attack
  - Small CM-discriminant
- All of these depend just on $p$ and the group order
- Fairly easy to avoid
- `safecurves.cr.yp.to`

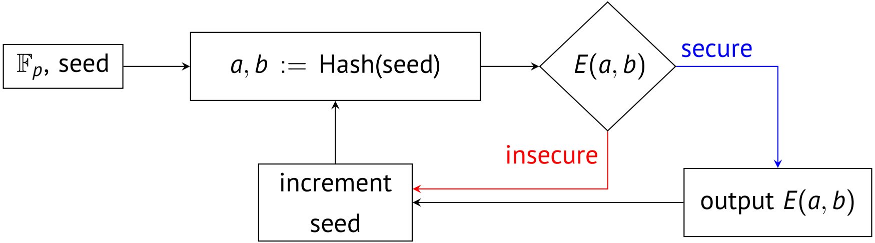
# ECDLP attacks

- Known attacks
- Unknown attacks
- Backdoors

# Timeline of standard curves

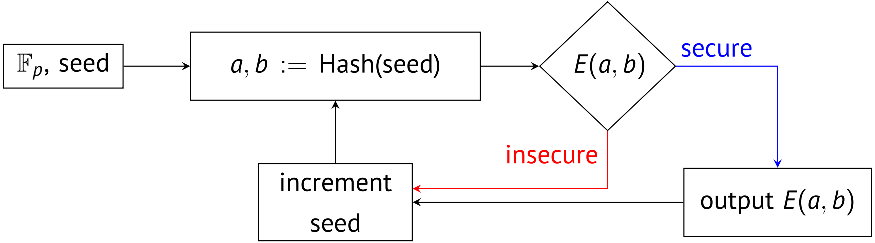


# Curve generation



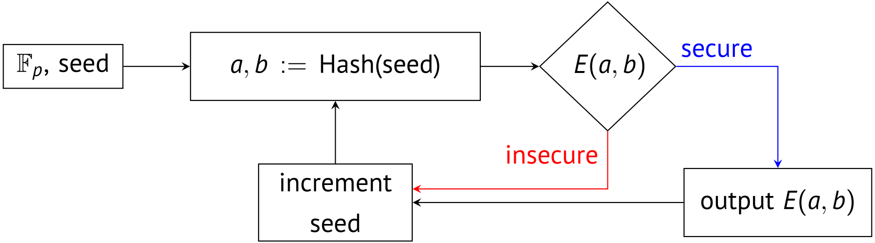
- verifiably pseudorandom: X9.62, SEC, Brainpool

# Curve generation



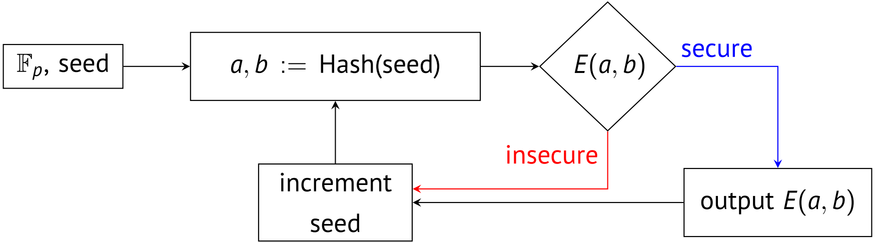
- verifiably pseudorandom: X9.62, SEC, Brainpool
- rigid: Curve25519, NUMS, MIRACL

# Curve generation



- verifiably pseudorandom: X9.62, SEC, Brainpool
- rigid: Curve25519, NUMS, MIRACL
- special/pairing-friendly: Bitcoin curve, BLS, BN, MNT

# Curve generation



- verifiably pseudorandom: X9.62, SEC, Brainpool
- rigid: Curve25519, NUMS, MIRACL
- special/pairing-friendly: Bitcoin curve, BLS, BN, MNT
- unknown/ambiguous origin: ANSSI FRP256v1, OSCCA SM2, GOST R

# ECDLP attacks

- Known attacks
- Unknown attacks
- Backdoors

# Backdoors

- Attacks that are known only to some party

# Backdoors

- Attacks that are known only to some party
- Incidents with backdoors in standards

# Backdoors

- Attacks that are known only to some party
- Incidents with backdoors in standards
  - Clipper chip

# Backdoors

- Attacks that are known only to some party
- Incidents with backdoors in standards
    - Clipper chip
    - Dual EC pseudorandom bit generator

# Backdoors

- Attacks that are known only to some party
- Incidents with backdoors in standards
  - Clipper chip
  - Dual EC pseudorandom bit generator
- Usual suspect: P-256 et al. (NIST + NSA)

# Backdoors

- Attacks that are known only to some party
- Incidents with backdoors in standards
  - Clipper chip
  - Dual EC pseudorandom bit generator
- Usual suspect: P-256 et al. (NIST + NSA)
  - $y^2 = x^3 - 3x + b$

# Backdoors

- Attacks that are known only to some party
- Incidents with backdoors in standards
  - Clipper chip
  - Dual EC pseudorandom bit generator
- Usual suspect: P-256 et al. (NIST + NSA)
  - $y^2 = x^3 - 3x + b$
  - $b = \sqrt{-27/\mathrm{Hash(seed)}}$

# Backdoors

- Attacks that are known only to some party
- Incidents with backdoors in standards
  - Clipper chip
  - Dual EC pseudorandom bit generator
- Usual suspect: P-256 et al. (NIST + NSA)
  - $y^2 = x^3 - 3x + b$
  - $b = \sqrt{-27/\mathrm{Hash(seed)}}$
  - seed = `c49d360886e704936a6678e1139d26b7819f7e90`

# Backdoors

- Attacks that are known only to some party
- Incidents with backdoors in standards
  - Clipper chip
  - Dual EC pseudorandom bit generator
- Usual suspect: P-256 et al. (NIST + NSA)
  - $y^2 = x^3 - 3x + b$
  - $b = \sqrt{-27/\mathrm{Hash(seed)}}$
  - seed = `c49d360886e704936a6678e1139d26b7819f7e90`
- Ambiguities in ECC standards

# Backdoors

- Attacks that are known only to some party
- Incidents with backdoors in standards
  - Clipper chip
  - Dual EC pseudorandom bit generator
- Usual suspect: P-256 et al. (NIST + NSA)
  - $y^2 = x^3 - 3x + b$
  - $b = \sqrt{-27/\mathrm{Hash}(\mathrm{seed})}$
  - seed = `c49d360886e704936a6678e1139d26b7819f7e90`
- Ambiguities in ECC standards
- Transparency is the key

# ECDLP attacks

- Known attacks ← safecurves.cr.yp.to
- Unknown attacks ← DiSSECT
- Backdoors ← DiSSECT

# ECDLP attacks

- Known attacks ← safecurves.cr.yp.to
- Unknown attacks ← DiSSECT
- Backdoors ← DiSSECT

# DiSSECT: simulations

- Idea:
    - compile a database of standard curves

# DiSSECT: simulations

- Idea:
  - compile a database of standard curves
  - compare standard curves to simulated ones

# DiSSECT: simulations

- Idea:
    - compile a database of standard curves
    - compare standard curves to simulated ones
    - look for weaknesses via statistical deviations

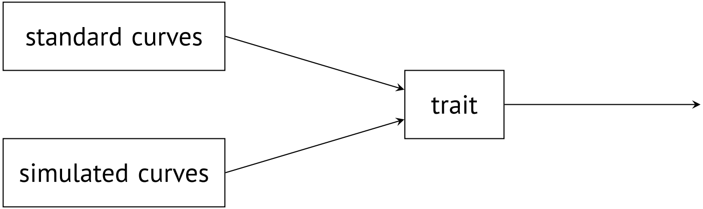
# DiSSECT: simulations

- Idea:
  - compile a database of standard curves
  - compare standard curves to simulated ones
  - look for weaknesses via statistical deviations

| | 256 bits | 224 bits | 192 bits | 160 bits | 128 bits |
|---|---|---|---|---|---|
| **X9.62**$_{\text{sim}}$ | 18 500 | 22 200 | 18 800 | 27 800 | 36 100 |
| **Brainpool**$_{\text{sim}}$ | 1 700 | 2 400 | 2 700 | 3 200 | 0 |
| **NUMS**$_{\text{sim}}$ | 100 | 100 | 200 | 300 | 0 |
| **Curve25519**$_{\text{sim}}$ | 100 | 0 | 400 | 300 | 0 |
| **Random** | 18 700 | 21 200 | 24 800 | 29 600 | 37 300 |

Simulation counts (>260k curves)

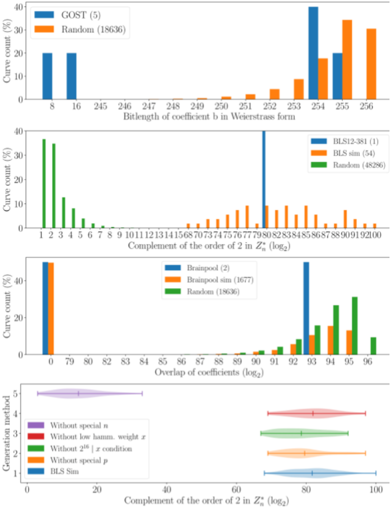
# DiSSECT: high level



standard curves

simulated curves

trait



# DiSSECT: traits

degrees
cm-field coefficients kn_factorization
degree graph twist nontrivial
full isogeny subgroup cofactor weight
containing ratio field factorization
coordinate discriminant order embedding
conductor
complement quadratic small torsion extensions twist_order
prime curve extension trace maximal torsion_extension
number cardinality least cm frobenius
relative polynomial
lower volcano prime-subgroup bitlength
conductors weierstrass upper basefield
class_number

# DiSSECT: interpretation

- Basic: manual eyeballing

# DiSSECT: interpretation

- Basic: manual eyeballing
- Advanced: automated outlier detection

# DiSSECT: interpretation

- Basic: manual eyeballing
- Advanced: automated outlier detection
  - Local: find outliers w.r.t. a single trait

# DiSSECT: interpretation

- Basic: manual eyeballing
- Advanced: automated outlier detection
  - Local: find outliers w.r.t. a single trait
  - Global: find outliers w.r.t. several traits at once

# DiSSECT: interpretation

- Basic: manual eyeballing
- Advanced: automated outlier detection
  - Local: find outliers w.r.t. a single trait
  - Global: find outliers w.r.t. several traits at once
- Some known properties inspired traits and suggest special treatment

# DiSSECT: interpretation

- Basic: manual eyeballing
- Advanced: automated outlier detection
  - Local: find outliers w.r.t. a single trait
  - Global: find outliers w.r.t. several traits at once
- Some known properties inspired traits and suggest special treatment
  - secp256k1 has $x(\frac{1}{2}G) < 2^{166}$ ... in fact, identical for secp224k1

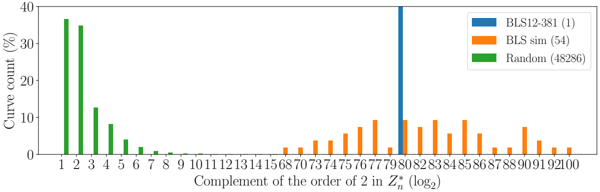
# DiSSECT: interpretation

- Basic: manual eyeballing
- Advanced: automated outlier detection
  - Local: find outliers w.r.t. a single trait
  - Global: find outliers w.r.t. several traits at once
- Some known properties inspired traits and suggest special treatment
  - secp256k1 has $x(\frac{1}{2}G) < 2^{166}$ . . . in fact, identical for secp224k1
  - Brainpool curves often have hex coefficients overlap

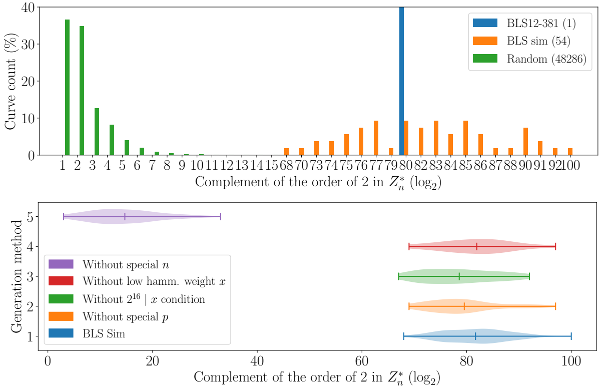
# DiSSECT: interpretation

- Basic: manual eyeballing
- Advanced: automated outlier detection
  - Local: find outliers w.r.t. a single trait
  - Global: find outliers w.r.t. several traits at once
- Some known properties inspired traits and suggest special treatment
  - secp256k1 has $x(\frac{1}{2}G) < 2^{166}$ ... in fact, identical for secp224k1
  - Brainpool curves often have hex coefficients overlap
- Comparison across standards often very valuable

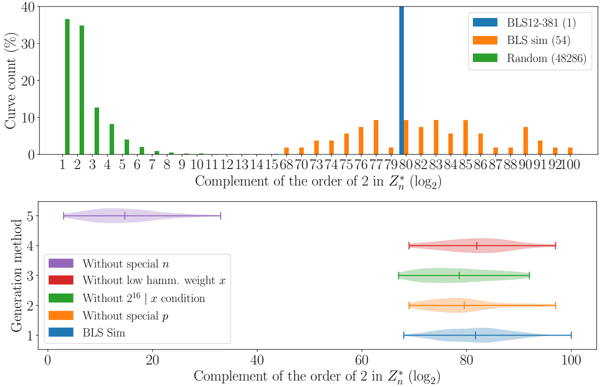
# DiSSECT: BLS12-381 findings



# DiSSECT: BLS12-381 findings

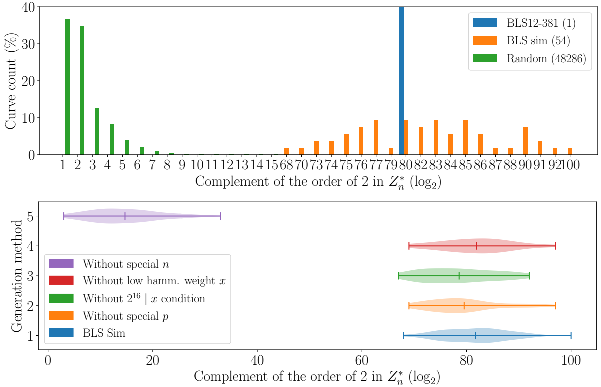


# DiSSECT: BLS12-381 findings



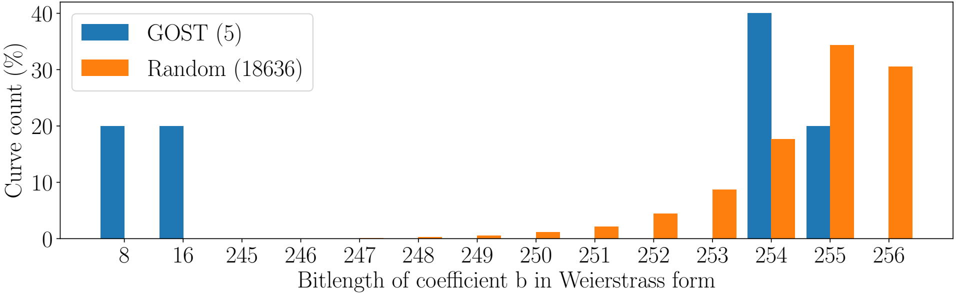
- Finding: $ord_n(2)$ is small

# DiSSECT: BLS12-381 findings



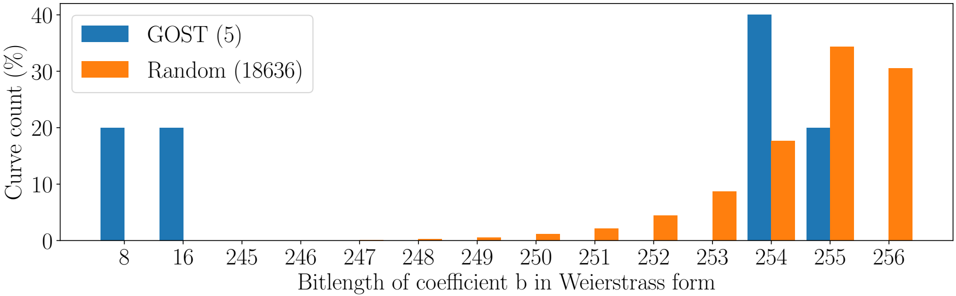
- Finding: $ord_n(2)$ is small
- Root cause: $\varphi(n) = x^2(x+1)(x-1)$ has no large factor

# DiSSECT: GOST findings



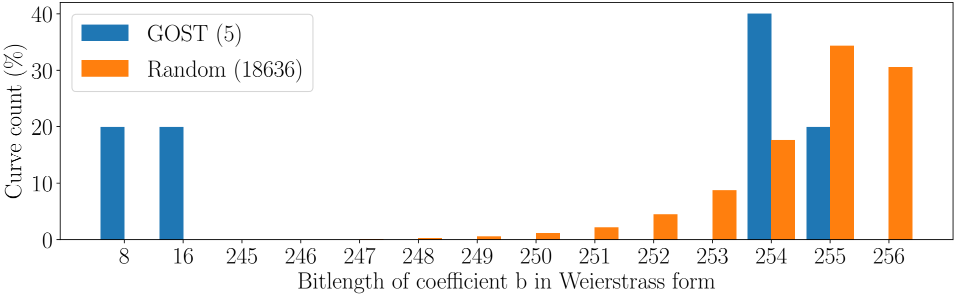
- Finding 1: `CryptoPro-A-ParamSet`, `CryptoPro-C-ParamSet` have small b coeffs

# DiSSECT: GOST findings



- Finding 1: `CryptoPro-A-ParamSet`, `CryptoPro-C-ParamSet` have small b coeffs
- Finding 2: `CryptoPro-B-ParamSet` has CM disc $-619$

# DiSSECT: GOST findings



- Finding 1: `CryptoPro-A-ParamSet`, `CryptoPro-C-ParamSet` have small b coeffs
- Finding 2: `CryptoPro-B-ParamSet` has CM disc $-619$
- Conclusion: these were generated in a special way

# Takeaways

- Systematic analysis required to trust non-transparent curves

# Takeaways

- Systematic analysis required to trust non-transparent curves
- DiSSECT: open source DB + analysis interface + visualisation

# Takeaways

- Systematic analysis required to trust non-transparent curves
- DiSSECT: open source DB + analysis interface + visualisation
- Found strange GOST and BLS properties

# Takeaways

- Systematic analysis required to trust non-transparent curves
- DiSSECT: open source DB + analysis interface + visualisation
- Found strange GOST and BLS properties
- Anyone can add a:

# Takeaways

- Systematic analysis required to trust non-transparent curves
- DiSSECT: open source DB + analysis interface + visualisation
- Found strange GOST and BLS properties
- Anyone can add a:
    - curve to be analyzed

# Takeaways

- Systematic analysis required to trust non-transparent curves
- DiSSECT: open source DB + analysis interface + visualisation
- Found strange GOST and BLS properties
- Anyone can add a:
  - curve to be analyzed
  - curve simulation method

# Takeaways

- Systematic analysis required to trust non-transparent curves
- DiSSECT: open source DB + analysis interface + visualisation
- Found strange GOST and BLS properties
- Anyone can add a:
  - curve to be analyzed
  - curve simulation method
  - trait to be applied to all curves

# Takeaways

- Systematic analysis required to trust non-transparent curves
- DiSSECT: open source DB + analysis interface + visualisation
- Found strange GOST and BLS properties
- Anyone can add a:
  - curve to be analyzed
  - curve simulation method
  - trait to be applied to all curves
- Let us leverage the large scale!

# Takeaways

- Systematic analysis required to trust non-transparent curves
- DiSSECT: open source DB + analysis interface + visualisation
- Found strange GOST and BLS properties
- Anyone can add a:
  - curve to be analyzed
  - curve simulation method
  - trait to be applied to all curves
- Let us leverage the large scale!
- WIP: pairing-friendly curves, clustering, entropy measurements,...

# Something ends, something begins

Questions and collaboration welcome!

Check out our tool and results at: https://dissect.crocs.fi.muni.cz/

CROCS

Centre for Research on
Cryptography and Security