Hi guys,

My name is Dimitri Koshelev. I am a researcher from Moscow and Paris. My field of science is elliptic curves and pairing-based cryptography.

I invented a quite efficient constant-time

hashing (that is without inversions and quadratic residuosity tests

) to some elliptic \mathbb{F}_{!p}

-curves E!: y^2 = x^3 + b

of j-invariant 0.

More precisely, the new hashing is applicable if Frobenius trace t := p+1 - |E(\mathbb{F}_{!p})|

is divided by 5 (i.e. there is a vertical \mathbb{F}_{!p^2}

-isogeny of degree 5

to E

) or Frobenius discriminant D := t^2 - 4p

is divided by 9 (i.e. there is a vertical \mathbb{F}_{!p}

-isogeny of degree 3

to E

).

My approach is similar to that for curves of j=1728 and the trick of Wahby-Boneh respectively. Moreover, the new hashing to BN512 is absolutely original from scientifique point of view.

The condition 5 | t is fulfilled, for example, for the Barreto-Naehrig curve BN512 (from the standard ISO15946-5). This curve may potentially be used in the near future. At the same time, 9 | D

for the curve BN256 from Article (early it was very popular in the industry).

Before me, there was only one known constant-time hashing to BN256 and BN512, namely SWU (Shallue-van de Woestijne-Ulas) hashing (see, e.g., El Mrabet, Joye, Guide to pairing-based cryptography, par. 8.4.2). However, it requires to perform 2 quadratic residuosity tests (QRT). If I'm not mistaken, the unique known simple constant-time implementation of QRT is 1 exponentiation in \mathbb{F}_{!p}

. But this is a very time-consuming operation.

In contrast, my hashing to BN512 in total contains only about 100 multiplications in \mathbb{F}_{!p}

(and the new hashing to BN256 is even more efficient).

It is worth noting that BN512 has no \mathbb{F}_{!p}

-isogenies of small degree from curves of j != 0. I checked that the smallest degree equals to 1291. Therefore the trick of Wahby-Boneh originally proposed for the curve BLS12-381 does not work for BN512.

In your opinion, is this a useful result ? Please let me know in order to collaborate if any of companies or startups continues to use BN256 in its products. In this case I can implement in one of programming languges the (very non-trivial) formulas of my hashing.

Best regards.