Some remarks on how to hash faster onto elliptic curves

Dmitrii Koshelev [1]

Computer sciences and networks department, Télécom Paris

Abstract. In this article we propose three optimizations of indifferentiable hashing onto (prime order subgroups of) ordinary elliptic curves over finite fields $\mathbb{F}_q$. One of them is dedicated to elliptic curves E provided that $q \equiv 11 \pmod{12}$. The other two optimizations take place respectively for the subgroups $\mathbb{G}_1$, $\mathbb{G}_2$ of some pairing-friendly curves. The performance gain comes from the smaller number of required exponentiations in $\mathbb{F}_q$ for hashing to $E(\mathbb{F}_q)$, $\mathbb{G}_2$ (resp. from the absence of necessity to hash directly onto $\mathbb{G}_1$). In particular, our results affect the pairing-friendly curve BLS12-381 (the most popular in practice at the moment) and the (unique) French curve FRP256v1 as well as almost all Russian standardized curves and a few ones from the draft NIST SP 800-186.

Key words: BLS12 family of pairing-friendly curves, clearing cofactor, indifferentiable hashing to elliptic curves, optimal ate pairings.

1 How to hash onto pairing-friendly curves

There is a lot of articles (including recent ones) on how to hash into or onto elliptic curves over finite fields. So, with your permission, we do not provide a detailed introduction in order to avoid repetition. Good surveys are represented in [1, §8], [2]. It is worth emphasizing that throughout this text we mean hashing indifferentiable from a random oracle (in the sense of [3, §2.2]).

Let E_1 be an ordinary pairing-friendly elliptic curve of embedding degree $k > 1$ over a finite field $\mathbb{F}_q$. Besides, let E_2 be a twist of E_1 of degree $d := \#\mathrm{Aut}(E_1)$ over the field $\mathbb{F}_{q^e}$, where $e := k/d \in \mathbb{N}$. As is customary, for a common prime divisor r of the orders $N_1 := \#E_1(\mathbb{F}_q)$ and $N_2 := \#E_2(\mathbb{F}_{q^e})$ denote by $\mathbb{G}_1 \subset E_1(\mathbb{F}_q)$ and $\mathbb{G}_2 \hookrightarrow E_2(\mathbb{F}_{q^e})$ the eigenspaces of the Frobenius endomorphism on $E_1[r] \subset E_1(\mathbb{F}_{q^k})$, associated with the eigenvalues $1, q$ respectively. Note that the condition $e \in \mathbb{N}$ is not automatically met, i.e., this is our assumption. It is claimed (e.g., in [1, Theorem 3.3.5]) that for any prime divisor $r \mid N_1$ there is always a unique non-trivial $\mathbb{F}_{q^e}$-twist E_2 (of degree d) such that $r \mid N_2$. By abuse of notation, we identify the order r subgroup $\mathbb{G}_2 \subset E_1(\mathbb{F}_{q^k})$ with its image under an $\mathbb{F}_{q^e}$-isomorphism $E_1 \xrightarrow{\sim} E_2$. Thus $\mathbb{G}_1 = E_1(\mathbb{F}_q)[r]$ and $\mathbb{G}_2 = E_2(\mathbb{F}_{q^e})[r]$. Besides, $d \in \{2, 4, 6\}$ and $d = 2$ if and only if $j(E_i) \neq 0, 1728$ (respectively, $d = 4$ iff $j(E_i) = 1728$ and $d = 6$ iff $j(E_i) = 0$).

This section explains how to hash onto $\mathbb{G}_2$ more efficiently and why we do not need to hash directly onto $\mathbb{G}_1$. In the first case, we significantly exploit the presence of clearing the cofactor $c_2 := N_2/r$. In the second one, on the contrary, clearing the cofactor $c_1 := N_1/r$ can be fully avoided. The fact is that *optimal ate pairings* $a\colon \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r \subset \mathbb{F}_{q^k}^*$ [1, Theorem 3.3.4] can

[1]web page: https://www.researchgate.net/profile/Dimitri-Koshelev
email: dimitri.koshelev@gmail.com

be painlessly (unlike $E_2(\mathbb{F}_{q^e}) \times \mathbb{G}_1$) extended to $\mathbb{G}_2 \times E_1(\mathbb{F}_q)$, at least in main pairing-based protocols.

At the moment, due to [4, Table 1] the curve BLS12-381 is a de facto standard in pairing-based cryptography. More generally, the *Barreto–Lynn–Scott family* with $k = 12$ and $d = 6$ (see, e.g., [5, §3.1]) possesses the parameters

$$r(z) = z^4 - z^2 + 1, \qquad q(z) = (z-1)^2 r(z)/3 + z.$$

By definition, BLS12-381 is generated by $z := -0xd201000000010000$ and hence

$$\lceil \log_2(-z) \rceil = 64, \qquad \lceil \log_2(r) \rceil = 255, \qquad \lceil \log_2(q) \rceil = 381.$$

Notice that $r \ll q$ in contrast to the *Barreto–Naehrig family* [1, Example 4.2].

Recall that almost all known hash functions $\mathcal{H}_i \colon \{0,1\}^* \to \mathbb{G}_i$ are the compositions $\mathcal{H}_i = [c_i'] \circ h_i \circ \eta_i$. Here $\eta_i \colon \{0,1\}^* \to S_i$ are hash functions to some finite sets, $h_1 \colon S_1 \to E_1(\mathbb{F}_q)$ and $h_2 \colon S_2 \to E_2(\mathbb{F}_{q^e})$ are just maps traditionally called *encodings*, and finally $c_i' \in \mathbb{N}$ such that $c_i \mid c_i'$, $r \nmid c_i'$. The scalar multiplication $[c_i']$ on the curve E_i is said to be *clearing cofactor*. Surprisingly, due to Fuentes-Castaneda et al. [6] it is more efficient to multiply points by scalars c_i' greater than c_i. The sets S_i are usually very simple, hence it is easy to combine η_i from existing hash functions $\{0,1\}^* \to \{0,1\}^\ell$ for $\ell \in \mathbb{N}$. The most complicated component of $\mathcal{H}_i$ is no doubt h_i, because its essence is based on high-dimensional algebraic geometry.

The majority of pairing-based protocols require a hash function to at most one group $\mathbb{G}_1$ or $\mathbb{G}_2$. Of course, any such protocol can be equivalently implemented for hashing to the other group. Without using point compression-decompression methods, elements of $\mathbb{G}_1$ (resp. $\mathbb{G}_2$) are obviously represented by $2\lceil \log_2(q) \rceil$ (resp. $2e\lceil \log_2(q) \rceil$) bits. Therefore the choice often depends on whether a hash value should be more compact than the second pairing argument or vice versa. Besides, there are rarely used protocols, for example the Scott identity-based key agreement [7], where both hash functions $\mathcal{H}_i$ are necessary. Thus the more cumbersome hashing to $\mathbb{G}_2$ cannot be replaced by hashing to $\mathbb{G}_1$ in all situations.

1.1 How not to hash onto $\mathbb{G}_1$

As far as we know, (non-degenerate) optimal ate pairings $a \colon \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r \subset \mathbb{F}_{q^k}^*$ are only used in today's real-world cryptography. The fact is that the corresponding Miller loop has the hypothetically smallest length $\approx \log_2(r)/\varphi(k)$, where φ is Euler's totient function. However it is more practical to take the whole group $E_1(\mathbb{F}_q)$ instead of $\mathbb{G}_1$. In this case, the pairing $a \colon \mathbb{G}_2 \times E_1(\mathbb{F}_q) \to \mu_r$ becomes degenerate, but this is not important. A similar trick is done in [8, §5] for the Tate pairing in the context of isogeny-based cryptography, where, on the contrary, $\mathbb{G}_2$ is replaced by $E_1(\mathbb{F}_{q^k})$ in our notation.

Indeed, first, the length of the Miller loop depends only on the order of $\mathbb{G}_2$. Second, if for points $P \in E_1(\mathbb{F}_q)$ and $Q \in \mathbb{G}_2$ we have $a(Q, P) = 1$, then a fortiori $a(Q, c_1'P) = a(Q, P)^{c_1'} = 1$. We stress that popular protocols (such as the Boneh–Franklin identity-based encryption [1, §1.6.4] or the aggregated BLS signature [9]) work correctly whether the order of P equals r or not. Nevertheless, it should be borne in mind that the *strong unforgeability property* (unlike the usual *existential* one) is not satisfied anymore as emphasized in [9, §5.2]. Finally, the complexity of computing $a(Q, P)$ remains the same as that of computing $a(Q, c_1'P)$, because P, $c_1'P$ are equally defined over $\mathbb{F}_q$.

In [10] an encoding $h_1 \colon \mathbb{F}_q^2 \to E_1(\mathbb{F}_q)$ is constructed for elliptic curves $E_1 \colon y^2 = x^3 + b$ (of j-invariant 0) provided that $\sqrt{b} \in \mathbb{F}_q$. There it is proved that h_1 is *admissible* in the sense of [3, Definition 4], which leads (in compliance with [3, Theorem 1]) to the indifferentiable hash function $h_1 \circ \eta_1$. Moreover, h_1 can be implemented in constant time of raising to some power $n_1 \in \mathbb{N}$ in the field $\mathbb{F}_q$ (not counting a few additions and multiplications). In particular, the encoding is applicable to the curve BLS12-381 for which $b = 4$ and $n_1 = (q-10)/27$.

Recall that the famous *(indirect) Wahby–Boneh encoding* h_{WB} [11, §4] (based on the *simplified SWU* one [3, §7]) is also valid for BLS12-381. It requires to extract one square root in $\mathbb{F}_q$, which for that curve is equivalent to raising in $\mathbb{F}_q$ to the power $n_2 := (q-3)/4 \in \mathbb{N}$. The hash function H_2 from [11, §5] twice applies h_{WB} in order to act as a random oracle. By the way, the other indifferentiable hash function H_3 is even slower than H_2 by virtue of [11, Figure 1].

To be exact, the Hamming weight $w(n_1) = 192$ and $w(n_2) = 228$. Denote by $\ell(n_i)$ the length of a shortest addition chain for n_i. In accordance with [12, §9.2.1] we obtain the inequalities

$$382 \leq \ell(n_1) \lesssim 419, \qquad 385 \leq \ell(n_2) \lesssim 422.$$

We cannot claim that these upper bounds are mathematically correct, because we omitted $o(1)$ in the original inequality. However, in any case, the sought bounds are very close (probably equal) to ours.

On the other hand, following the sliding window method [12, §9.1.3] (with $k = 5$), we explicitly derive in Magma [13] an addition chain for n_1 (resp. n_2) whose the length equals 449 (resp. 458). Curiously, a similar chain for n_2 of the same length 458, obtained by means of more advanced methods, appears in the optimized library blst [14]. Thus the encoding h_{WB} applied twice is much slower than the one h_1 applied once. Indeed, $2 \cdot 458 - 449 = 467$ is a significant amount of multiplications in $\mathbb{F}_q$ that can be eliminated by giving priority to h_1.

1.2 How to hash onto $\mathbb{G}_2$

To our knowledge, optimal ate pairings do not have a natural extension to $E_2(\mathbb{F}_{q^e}) \times \mathbb{G}_1$. Conversely, (non-degenerate) *twisted optimal ate pairings* [1, Theorem 3.3.8] of the form $\mathbb{G}_1 \times \mathbb{G}_2 \to \mu_r$ are readily extended to $\mathbb{G}_1 \times E_2(\mathbb{F}_{q^e})$. But, unfortunately, for them the Miller loop is of a larger length than for (usual) optimal ate pairings. It is generally recognized that a pairing is a more laborious operation than an elliptic curve scalar multiplication. Therefore reducing the Miller loop seems a better solution than avoiding the multiplication by c_2'.

For the sake of convenience, introduce so-called *tensor multiplication* of any two maps $h \colon S \to G$, $g \colon T \to G$ from sets S, T to the same group $(G, +)$:

$$h \otimes g \colon S \times T \to G \qquad (s,t) \mapsto h(s) + g(t).$$

We know (e.g., from [1, Theorem 2.11]) that $E_2(\mathbb{F}_{q^e}) \simeq \mathbb{Z}/(mr) \times \mathbb{Z}/\ell$, where $\ell \mid m$ and $m\ell = c_2$. Pick any independent points $P_0, P_1 \in E_2(\mathbb{F}_{q^e})$ of orders m and ℓ respectively. The independency means that $P_1 \in E_2(\mathbb{F}_{q^e}) \setminus \langle P_0 \rangle$ if $\ell > 1$, and $P_1 = (0:1:0)$ if $\ell = 1$. Consider the set $V := [0, m) \times [0, \ell)$ and the maps

$$g \colon V \to E_2(\mathbb{F}_{q^e}) \qquad (v_0, v_1) \mapsto v_0 P_0 + v_1 P_1,$$

$$F\colon \mathbb{F}_{q^e} \times V \to \mathbb{G}_2 \qquad F := [c_2'] \circ (h_2 \otimes g).$$

For the next theorem we need the notions of (B-)*well-distributed encoding* [15, Definitions 5] and (ϵ-)*regular map* [15, Definition 3] (with respect to the uniform distribution on its domain).

Theorem 1. *Assume that $h_2\colon \mathbb{F}_{q^e} \to E_2(\mathbb{F}_{q^e})$ is a B-well-distributed encoding (for $B \in \mathbb{R}_{>0}$). Then the map F is ϵ-regular, where $\epsilon := B\sqrt{r/q^e}$. As a result, ϵ is negligible whenever $e > 1$.*

This is an immediate consequence of [15, Corollary 1] and [3, Lemma 13].

Note that F is a *samplable map* (in the sense of [3, Definition 4]) if, as is often the case, h_2 enjoys a large image, that is $\#\mathrm{Im}(h_2) = \Theta(q^e)$. Indeed, this property follows from [3, Lemma 13] and [15, Algorithm 1]. Eventually, we establish

Corollary 1. *The map F is admissible.*

Corollary 2. *If a hash function $\eta\colon \{0,1\}^* \to \mathbb{F}_{q^e}$ is indifferentiable from a random oracle, then the hash function $[c_2'] \circ h_2 \circ \eta\colon \{0,1\}^* \to \mathbb{G}_2$ (denoted by* H_4 *in* [11, §5]*) is so.*

Proof. Take another random oracle $\theta\colon \{0,1\}^* \to V$. Therefore the functions $(\eta,\theta)(s) := \big(\eta(s),\theta(s)\big)$ and hence $F \circ (\eta,\theta)\colon \{0,1\}^* \to \mathbb{G}_2$ also act as a random oracle (the second fact is [3, Theorem 1]). Finally, obviously, $\mathrm{H}_4 = F \circ (\eta,\theta)$. $\square$

For the BLS12-381 curve $E_2\colon y^2 = x^3 + 4(1+i)$ (where $i := \sqrt{-1} \notin \mathbb{F}_q$) in the role of h_2 the article [11, §5] proposes the Wahby–Boneh encoding. However that article does not notice the indifferentiability of H_4. By the way, the other (indifferentiable) hash functions H_5, H_6 are even slower than H_4 by virtue of [11, Figure 1].

2 How to hash onto $E(\mathbb{F}_q)$ if $q \equiv 11 \pmod{12}$

Hash functions to classical (i.e., non-pairing-friendly) elliptic curves have become more and more in demand. Indeed, according to [16, Table I] they are actively used in many PAKE (Password Authenticated Key Exchange) protocols. Several years ago CFRG (Crypto Forum Research Group) conducted the PAKE selection process [17] in which the protocols CPace [18] and OPAQUE [19] won. Besides, hashing to elliptic curve is necessary for some blind signatures (such as in [20, §3.3]), which serve as a basis, e.g., for electronic voting schemes.

Let us freely utilize notions arisen in previous sections. Consider an elliptic curve $E\colon y^2 = x^3 + ax + b$ defined over a finite field $\mathbb{F}_q$. Under the condition $q \equiv 2 \pmod 3$ (resp. $j(E) \neq 0, 1728$), *Icart's encoding* h_I [21] (resp. the simplified SWU one h_{sSWU}) is available. In accordance with [21, Lemma 4], [3, Lemma 6] for any $P \in E(\mathbb{F}_q)$ we have $\#h_I^{-1}(P) \leqslant 4$ and $\#h_{sSWU}^{-1}(P) \leqslant 8$. In fact, if an implementation of h_{sSWU} takes into account the sign of the y-coordinate, then $\#h_{sSWU}^{-1}(P) \leqslant 4$. At the same time, by virtue of [22, §5] the encoding h_I (resp. h_{sSWU}) is B-well-distributed with $B = 13$ (resp. $B = 53$) at least for q of a cryptographic size. Applying [15, Corollary 1], we thus get

Lemma 1. *Suppose that $q \equiv 2 \pmod 3$ and $j(E) \neq 0, 1728$. Then the map $F := h_I \otimes h_{sSWU}\colon \mathbb{F}_q^2 \to E(\mathbb{F}_q)$ is ϵ-regular for the negligible value $\epsilon := 26\sqrt{N}/q$, where $N := \#E(\mathbb{F}_q)$.*

From now on we assume in addition that $q \equiv 3 \pmod 4$. Obviously,

$$q \equiv 2 \pmod 3,\ q \equiv 3 \pmod 4 \qquad \Leftrightarrow \qquad q \equiv 11 \pmod{12}.$$

For the sake of compactness, we put $e := (q+1)/4$ and $k := (q+1)/12$. Notice that for $Z = n/d$ such that $n, d \in \mathbb{F}_q^*$ we obtain

$$z := Z^k = n^k \cdot d^{q-1-k} = n^k \cdot d^{(11q-13)/12} = nd^9 \cdot (nd^{11})^{(q-11)/12}, \qquad z^6 = Z^{(q+1)/2} = \left(\frac{Z}{q}\right) Z,$$

where $\left(\frac{Z}{q}\right)$ is the Legendre symbol. In particular, $z = \sqrt[6]{Z}$ whenever Z is a quadratic residue in $\mathbb{F}_q$.

Given $(t, s) \in \mathbb{F}_q^2$ we need to evaluate $h_I(t)$ and $h_{sSWU}(s)$. As is known, separately each of these points can be computed in constant time of one exponentiation in $\mathbb{F}_q$ (the case of h_{sSWU} see in [11, §4.2]). Let's show that this is also possible simultaneously for the two points (and hence for $F(t, s)$). The only cumbersome part of h_I (resp. h_{sSWU}) consists in the exponentiation $\sqrt[3]{f} = f^{(2q-1)/3}$ (resp. $\pm g^e$ such that $(g^e)^2 = \left(\frac{g}{q}\right) g$), where

$$f := \left(\frac{3a - t^4}{6t}\right)^2 - b - \frac{t^6}{27}, \qquad g := -\frac{b}{a}\left(1 + \frac{1}{s^4 - s^2}\right).$$

Evidently, $\sqrt[3]{f}$ is the unique cubic root of f in $\mathbb{F}_q$ and for our purpose it is sufficient to find g^e up to a sign. For the sake of simplicity, let us exclude from consideration the zeros and poles of the functions f, g. As usual, they can be processed individually.

We suggest to act in a similar way as in [23, §3], that is for $Z := f^2 g^3$ to compute $z = Z^k$ (almost $\sqrt[6]{Z}$) instead of separate computing $\sqrt[3]{f}$ and $\pm g^e$ (almost $\sqrt{g}$). Note that

$$z = f^{(q+1)/6} \cdot g^e = \left(\frac{f}{q}\right) \sqrt[3]{f} \cdot g^e, \qquad z^2 = \sqrt[3]{f^2} \cdot \left(\frac{g}{q}\right) g.$$

Introducing the auxiliary notation $\theta := fg/z^2$, we get the equalities

$$\sqrt[3]{f} = \frac{\left(\frac{g}{q}\right) fg}{z^2} = \left(\frac{g}{q}\right)\theta, \qquad g^e = \frac{z}{\left(\frac{f}{q}\right)\sqrt[3]{f}} = \frac{z}{\left(\frac{fg}{q}\right)\theta}.$$

We see that $\theta^3 = \left(\frac{g}{q}\right) f$ and $z^6 = \left(\frac{g}{q}\right) Z$. Therefore the symbol $\left(\frac{g}{q}\right)$ can be determined for free. More formally,

$$\left(\sqrt[3]{f},\ \pm g^e\right) = \begin{cases} \left(\theta,\ z/\theta\right) & \text{if} \qquad \theta^3 = f, \ \text{i.e.,}\ z^6 = Z, \\ \left(-\theta,\ z/\theta\right) & \text{otherwise.} \end{cases}$$

Bearing in mind the formula above for $(n/d)^k$ without the inversion operation, we emphasize again that

Remark 1. *The map F (in contrast to $h_I^{\otimes 2}$ and $h_{sSWU}^{\otimes 2}$) can be computed in constant time of one exponentiation in $\mathbb{F}_q$.*

Of course, by analogy with [13], given q it is not difficult to derive explicit short addition chains for raising to the power k. Besides, F is a samplable map due to [15, Algorithm 1], which eventually leads to

Corollary 3. *The map* $F\colon \mathbb{F}_q^2 \to E(\mathbb{F}_q)$ *is admissible.*

Remark 1 is still valid when h_{sSWU} is replaced by any encoding implementable with the cost of extracting one square root in $\mathbb{F}_q$. We chose h_{sSWU}, because it is the most universal among such encodings known in the literature. In particular, this encoding is relevant even if N is a prime (that is the cofactor equals 1), which is the case for many classical elliptic curves. Note that for $q \equiv 11 \pmod{12}$ curves of j-invariants 0, 1728 are supersingular in compliance with [12, §24.2.1.c]. Since such curves pose special challenges for security by virtue of [1, Remark 2.22], the map h_{sSWU} does not have restrictions in the current context.

There is a lot of standardized elliptic curves over fields $\mathbb{F}_q$ such that $q \equiv 11 \pmod{12}$. It is readily checked that this condition is fulfilled, e.g., for the French curve FRP256v1 [24], for the curves P-192, P-384, and Curve448-Goldilocks from NIST SP 800-186 [25, §4.2.1] as well as for all Russian curves [26, Appendix B] except for id-GostR3410-2001-CryptoPro-B-ParamSet. Possibly, Remark 1 can be generalized to the case $q \equiv 2 \pmod{3}$, $q \equiv 5 \pmod{8}$ when a square root is still expressed via one exponentiation (see, e.g., [2, Appendix I.2]). However we did not find standardized curves over such fields, hence we decided to stop in order not to complicate the text.

References

[1] El Mrabet N., Joye M., *Guide to Pairing-Based Cryptography*, Cryptography and Network Security Series, Chapman and Hall/CRC, New York, 2017.

[2] Faz-Hernandez A. et al., *Hashing to elliptic curves*, https://datatracker.ietf.org/doc/draft-irtf-cfrg-hash-to-curve, 2021.

[3] Brier E. et al., "Efficient indifferentiable hashing into ordinary elliptic curves", Advances in Cryptology — CRYPTO 2010, LNCS, **6223**, ed. Rabin T., Springer, Berlin, Heidelberg, 2010, 237–254.

[4] Sakemi Y., Kobayashi T., Saito T., Wahby R. S., *Pairing-friendly curves*, https://datatracker.ietf.org/doc/draft-irtf-cfrg-pairing-friendly-curves, 2021.

[5] Budroni A., Pintore F., "Efficient hash maps to $\mathbb{G}_2$ on BLS curves", *Applicable Algebra in Engineering, Communication and Computing*, 2020, 1–21.

[6] Fuentes-Castaneda L., Knapp E., Rodríguez-Henríquez F., "Faster hashing to $\mathbb{G}_2$", Selected Areas in Cryptography. SAC 2011, LNCS, **7118**, eds. Miri A., Vaudenay S., Springer, Berlin, Heidelberg, 2012, 412–430.

[7] Scott M., *Authenticated ID-based key exchange and remote log-in with simple token and PIN number*, https://eprint.iacr.org/2002/164, 2002.

[8] Pereira G., Doliskani J., Jao D., "x-only point addition formula and faster compressed SIKE", *Journal of Cryptographic Engineering*, **11**:1 (2021), 57–69.

[9] Boneh D., Gorbunov S. et al., *BLS signatures*, https://datatracker.ietf.org/doc/draft-irtf-cfrg-bls-signature, 2020.

[10] Koshelev D., *Indifferentiable hashing to ordinary elliptic* $\mathbb{F}_q$*-curves of* $j = 0$ *with the cost of one exponentiation in* $\mathbb{F}_q$, https://eprint.iacr.org/2021/301, accepted in Designs, Codes and Cryptography, 2021.

[11] Wahby R. S., Boneh D., “Fast and simple constant-time hashing to the BLS12-381 elliptic curve”, *IACR Transactions on Cryptographic Hardware and Embedded Systems*, **2019**:4 (2019), 154–179.

[12] Cohen H. et al., *Handbook of Elliptic and Hyperelliptic Curve Cryptography*, Discrete Mathematics and Its Applications, **34**, Chapman and Hall/CRC, New York, 2005.

[13] Koshelev D., *Magma code*, `https://github.com/dishport/Some-remarks-on-how-to-hash-faster-onto-elliptic-curves`, 2021.

[14] Supranational, *blst/src/sqrt-addchain.h*, `https://github.com/supranational/blst/blob/c76b5ac69a0044432d16cfd2cce60c93c8b01872/src/sqrt-addchain.h`, 2020.

[15] Tibouchi M., Kim T., “Improved elliptic curve hashing and point representation”, *Designs, Codes and Cryptography*, **82**:1–2 (2017), 161–177.

[16] Hao F., *Prudent practices in security standardization*, `https://eprint.iacr.org/2021/839`, 2021.

[17] Crypto Forum Research Group (CFRG), *PAKE selection process*, `https://github.com/cfrg/pake-selection`, 2020.

[18] Abdalla M., Haase B., Hesse J., *CPace, a balanced composable PAKE*, `https://datatracker.ietf.org/doc/draft-irtf-cfrg-cpace/02`, 2021.

[19] Krawczyk H., Bourdrez D., Lewi K., Wood C. A., *The OPAQUE asymmetric PAKE protocol*, `https://www.ietf.org/id/draft-irtf-cfrg-opaque-06.html`, 2021.

[20] Abe M., Okamoto T., “Provably secure partially blind signatures”, Advances in Cryptology — CRYPTO 2000, LNCS, **1880**, eds. Bellare M., Springer, Berlin, Heidelberg, 2000, 271–286.

[21] Icart T., “How to hash into elliptic curves”, Advances in Cryptology — CRYPTO 2009, LNCS, **5677**, eds. Halevi S., Springer, Berlin, Heidelberg, 2009, 303–316.

[22] Farashahi R. R. et al., “Indifferentiable deterministic hashing to elliptic and hyperelliptic curves”, *Mathematics of Computation*, **82**:281 (2013), 491–512.

[23] Koshelev D., “Faster point compression for elliptic curves of j-invariant 0”, `https://eprint.iacr.org/2020/010`, *Mathematical Aspects of Cryptography*, **12**:4 (2021), 27–35.

[24] Agence Nationale de la Sécurité des Systèmes d’Information (ANSSI), *Avis relatif aux paramètres de courbes elliptiques définis par l’Etat français*, `https://www.legifrance.gouv.fr/jorf/id/JORFTEXT000024668816`, 2011.

[25] Chen L., Moody D., Regenscheid A., Randall K., *Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters (Draft NIST Special Publication 800-186)*, `https://csrc.nist.gov/publications/detail/sp/800-186/draft`, 2019.

[26] Alekseev E. K., Nikolaev V. D., Smyshlyaev S. V., “On the security properties of Russian standardized elliptic curves”, *Mathematical Aspects of Cryptography*, **9**:3 (2018), 5–32.