Optimal TNFS-secure pairings on elliptic curves with composite embedding degree

Georgios Fotiadis[1] and Chloe Martindale[2] ⋆

[1] SnT, University of Luxembourg, Luxembourg
georgios.fotiadis@uni.lu
[2] Technische Universiteit Eindhoven, the Netherlands
chloemartindale@gmail.com

Abstract. In this paper we present a comprehensive comparison between pairing-friendly elliptic curves, considering different curve forms and twists where possible. We define an additional measure of the efficiency of a parametrized pairing-friendly family that takes into account the number field sieve (NFS) attacks (unlike the ρ-value). This measure includes an approximation of the security of the discrete logarithm problem in $\mathbb{F}_{p^k}^*$, computed via the method of Barbulescu and Duquesne [4]. We compute the security of the families presented by Fotiadis and Konstantinou in [14], compute some new families, and compare the efficiency of both of these with the (adjusted) BLS, KSS, and BN families, and with the new families of [20]. Finally, we recommend pairing-friendly elliptic curves for security levels 128 and 192.

Keywords: Optimal ate pairing, twists of elliptic curves, jacobian coordinates, TNFS-secure, SexTNFS.

1 Introduction

Pairings first appeared in 1940 when André Weil showed that there is a way to map points of order r on a supersingular elliptic curve to an element of order r in a finite field; his map became known as the *Weil pairing*. In 1986, Victor Miller [25] gave an algorithm that computes the Weil pairing efficiently, and in 1993, Menezes, Okamoto and Vanstone [24] applied Miller's method to the elliptic curve discrete logarithm problem (ECDLP) for supersingular elliptic curves. They reduced ECDLP for supersingular elliptic curves to the discrete logarithm problem in a finite field (DLP), giving a subexponential attack now known as the MOV-attack. This attack was followed by the more general FR-reduction [16], which can be applied to ordinary elliptic curves (and higher-dimensional abelian

⋆ Author list in alphabetical order; see https://www.ams.org/profession/leaders/culture/CultureStatement04.pdf. Georgios Fotiadis was supported by European Union's Horizon 2020 research and innovation programme under grant agreement No. 779391 (FutureTPM). Chloe Martindale was supported by the Netherlands Organisation for Scientific Research (NWO) under CHIST-ERA USEIT (grant number 651.002.004). Date of this document: August 22, 2019.

varieties) when using a variant of the Weil pairing called the *Tate pairing*. In the early 2000s, several authors presented efficient pairing-based protocols (see e.g. [8,9,21]) which are now the backbone of privacy-related cryptosystems, the security of which relies on the aforementioned cryptanalysis. More recently, pairings have started being deployed in the marketplace, for example in the Elliptic Curve Direct Anonymous Attestation (ECDAA) protocol that is embedded in the current version of the Trusted Platform Module (TPM), namely TPM2.0 [19].

1.1 Pairings on elliptic curves

For the introductory material on pairings and attacks in this section, Section 1.2, and Section 2.1, we follow [15, Section 1.1], [15, Section 1.2], and [15, Section 2.1] respectively.

Let $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ be cyclic groups of prime order r and assume that the DLP is intractable in all three groups. An *abstract pairing* is a bilinear, non-degenerate, efficiently computable map of the form:

$$\widehat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T. \tag{1}$$

When $\mathbb{G}_1 \neq \mathbb{G}_2$ the pairing is called *asymmetric*.

Let E be an ordinary elliptic curve defined over a prime field $\mathbb{F}_p$ and let r be largest prime such that $r|\#E(\mathbb{F}_p)$. We define the *embedding degree* k of E to be the minimal integer k for which all the r-th roots of unity are contained in $\mathbb{F}_{p^k}$. For all pairings on elliptic curves that are currently used in cryptography, the groups $\mathbb{G}_1$ and $\mathbb{G}_2$ are r-order subgroups of $E(\mathbb{F}_{p^k})$ and the group $\mathbb{G}_T$ is the subgroup $\mu_r \subseteq \mathbb{F}^*_{p^k}$ of r^{th} roots of unity. That is, a pairing of elliptic curves is a map:

$$\widehat{e} : E(\mathbb{F}_{p^k})[r] \times E(\mathbb{F}_{p^k})[r] \to \mu_r \subset \mathbb{F}^*_{p^k}.$$

The most widely used pairings on ordinary elliptic curves can be efficiently computed using variations of Miller's algorithm [25].

1.2 Attacks on pairings

For the curves that we consider, the best attack on ECDLP in an r-order subgroup of $E(\mathbb{F}_{p^k})$ is Pollard's rho algorithm, which has complexity $O(\sqrt{r})$. The complexity of DLP in the multiplicative group $\mathbb{F}^*_{p^k}$ depends both on the construction of k and of p. In the case of most pairing-friendly curves, the primes p and r are large (at least 256 bits) and are derived from the evaluation of polynomials; we say that such primes are of *special form*.

When the embedding degree k is prime, the asymptotic complexity of DLP in $\mathbb{F}^*_{p^k}$ is $L_{p^k}[1/3, 1.923]$, due to the number field sieve (NFS) method. For composite embedding degrees, Kim and Barbulescu's [23] improvements on the tower number field sieve (TNFS) method have reduced the complexity of DLP in $\mathbb{F}^*_{p^k}$ from $L_{p^k}[1/3, 1.923]$ to $L_{p^k}[1/3, 1.526]$ (in the most extreme cases). The concrete complexity of the NFS method for a given example can be computed using Barbulescu and Duquesne's method [4]. These new improvements have immediate

consequences on the selection of the extension fields $\mathbb{F}_{p^k}$. A summary of (new) recommendations of pairing-friendly elliptic curves, for many different embedding degrees, that are resistant against the new TNFS attacks is presented in [14]. The security of these recommendations was approximated via asymptotics.

1.3 Our contributions

We present new candidates for pairing-friendly families for both security levels 128 and 192. We also introduce a new measure, the τ_n-value, for the efficiency of a pairing-friendly family that takes into account the latest attacks (unlike the ρ-value). We compare the candidate families and recommend two families for security level 128 together with a choice of elliptic curve in that family, which should perform about 12% faster than the fastest pairing-friendly elliptic curve that was previously available the literature. Finally, we recommend two efficient pairing-friendly elliptic curves for security level 192.

Acknowledgments

We would especially like to thank Prof. Sylvain Duquesne for taking the time to discuss with us in detail his work with Barbulescu on computing the complexity of attacks on pairings, and for sharing his code with us. We would also like to thank Rémi Clarisse for useful discussions.

2 Preliminaries

We use the method of Brezing and Weng [10] for generating pairing-friendly families of elliptic curves, and in order to make concrete recommendations we use the method of Barbulescu and Duquesne [4] to compute the security of the target group. We briefly summarize these here for the convenience of the reader.

2.1 Pairing-Friendly Curves

For the construction of pairing-friendly elliptic curves we use *complete polynomial families*, introduced by Brezing and Weng in [10]. For a given embedding degree $k > 0$ the elliptic curve parameters p, t and r are described as polynomials $p(x), t(x), r(x) \in \mathbb{Q}[x]$ respectively such that

$$4p(x) - t(x)^2 = Dy(x)^2,$$

where $D > 0$ is the CM discriminant and $y(x) \in \mathbb{Q}[x]$. These polynomials must also satisfy the relation

$$\Phi_k(t(x) - 1) \equiv 0 \bmod r(x),$$

where $\Phi(x)$ is the k^{th} cyclotomic polynomial. Additionally, these polynomials satisfy $p(x) + 1 - t(x) \equiv 0 \bmod r(x)$, which ensures that the order of the elliptic curve has a polynomial representation as $\#E(\mathbb{F}_{p(x)}) = h(x)r(x)$.

We can generate elliptic curve parameters by evaluating the polynomial family at some integer u such that $r = r(u)$ and $p = p(u)$ are both prime and $t = t(u) \leq 2\sqrt{p}$ (cf. Hasse bound). The polynomials $p(x), t(x), r(x)$ from our constructions are *integer-valued*, which we define to mean that there exist infinitely many $u \in \mathbb{Z}$ for which these polynomials produce integer values.

2.2 Efficient Pairings

To our knowledge, the most efficient asymmetric pairing in the literature is the *optimal ate pairing*, which was introduced by Vercauteren in [26]. In this case we set:

$$\mathbb{G}_1 = E(\overline{\mathbb{F}_p})[r] \cap \ker(\pi_p - [1]) \quad \text{and} \quad \mathbb{G}_2 = E(\overline{\mathbb{F}_p})[r] \cap \ker(\pi_p - [p]),$$

where π_p denotes the p-power Frobenius endomorphism on E. Note that $\mathbb{G}_1 = E(\mathbb{F}_p)[r]$ and $\mathbb{G}_2 \subseteq E(\mathbb{F}_{p^k})[r]$.

Consider the $\varphi(k)$-dimensional lattice L (spanned by the rows):

$$L = \begin{pmatrix} r & 0\,0 \ldots 0 \\ -p & 1\,0 \ldots 0 \\ -p^2 & 0\,1 \ldots 0 \\ \vdots & \vdots\,\vdots\,\ddots\,\vdots \\ -p^{\varphi(k)-1} & 0\,0 \ldots 1 \end{pmatrix} \tag{2}$$

and let $V = [c_0, c_1, \ldots, c_{\varphi(k)-1}]$ be the shortest vector of this lattice. The optimal ate pairing is defined as the bilinear, non-degenerate map $\widehat{a} : \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r \subset \mathbb{F}_{p^k}^*$ given by

$$(Q, P) \mapsto \left[\prod_{i=0}^{\varphi(k)-1} f_{c_i,Q}^{p^i}(P) \cdot \underbrace{\prod_{i=0}^{\varphi(k)-2} h_{[s_{i+1}]Q,[c_i p^i]Q}(P)}_{H} \right]^{\frac{p^k-1}{r}}, \tag{3}$$

where for two points R, S on the curve E, $h_{R,S}$ is the rational function with divisor $(R) + (S) - (S + R) - P_\infty$ and the values s_i are obtained by the relation:

$$s_i = \sum_{j=i}^{\varphi(k)-1} c_j p^j.$$

By [26] this choice of the coordinates c_i ensures the non-degeneracy property of the above pairing. This pairing also has the following nice properties (see [26] for details and proofs):

– For every $c_i < 0$, we have $f_{c_i,Q} = 1/(f_{-c_i,Q} v_{[c_i]Q})$, where $v_{[c_i]Q}$ is the vertical line passing through the point $[c_i]Q$.
– For every $a, b \in \mathbb{Z}_{\geq 0}$ we have $f_{ab,Q} = f_{a,Q}^b f_{b,[a]Q}$.
– It is trivial to see that $f_{0,Q} = f_{1,Q} = f_{-1,Q} = 1$.

The optimal ate pairing can be computed efficiently via via Miller's algorithm (Algorithm 1). Furthermore, on a fundamental level the pairing computation can be broken down into arithmetic operations in $\mathbb{F}_p$, which is a useful way to compare the efficiency of different choices for the input to Algorithm 1.

We follow this method to compare our recommendations with earlier suggestions; that is, by giving the complexity of computing one pairing as a number of multiplications in $\mathbb{F}_p$. We can then compare between the choices with different sizes of finite field by counting clock cycles for a generic algorithm (such as Montgomery multiplication) for $\mathbb{F}_p$-multiplication.

As is standard in the pairing literature, we do not count $\mathbb{F}_p$-additions as the total cost is dominated by multiplications.

Algorithm 1 Miller's algorithm

Input: $P \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$, $V = [c_0, c_1, \ldots, c_{\varphi(k)-1}]$.
Output: $\widehat{a}(Q, P)$.
1: $F \leftarrow 1$;
2: **for** $j = 0$ to $\varphi(k) - 1$ **do**
3: $\quad n \leftarrow \lfloor \log_2 c_j \rfloor$; $f \leftarrow 1$; $R \leftarrow Q$; $(T_n, T_{n-1}, T_{n-2}, \ldots, T_1, T_0) \leftarrow \text{binary}(c_j)$;
4: $\quad$ **for** $i = n - 1$ to 0 **do**
5: $\quad\quad f \leftarrow f^2 \cdot h_{R,R}(P)$
6: $\quad\quad R \leftarrow 2R$
7: $\quad\quad$ **if** $T_i \neq 0$ **then**
8: $\quad\quad\quad f \leftarrow f \cdot h_{R,T_i \cdot Q}(P)$
9: $\quad\quad\quad R \leftarrow R + T_i \cdot Q$
10: $\quad$ **if** $c_j < 0$ **then**
11: $\quad\quad f \leftarrow 1/f$
12: $\quad F \leftarrow F \cdot f$
13: $F \leftarrow F \cdot H$ (cf. (3)).
14: **return** $F^{\frac{p^k-1}{r}}$

The computation of the cost of Algorithm 1 is typically split into two parts: the *Miller loop*, defined as the cost of steps 3-12, for each coordinate c_j of the shortest vector V, and the *final exponentiation*, defined as the cost of raising an element in $\mathbb{F}_{p^k}$ to the power of $\frac{p^k-1}{r}$. Note that if two divides k, then the expensive-looking inversion in Step 11 can be replaced by exponentiation by $p^{k/2}$ [2]. This is just conjugation in $\mathbb{F}_{p^k}/\mathbb{F}_{p^{k/2}}$, so is free. For computing the number of $\mathbb{F}_p$-multiplications in the Miller loop, we followed the same method as Guillevic, Masson, and Thomé [20].

Final exponentiation. Raising an element $f \in \mathbb{F}_{p^k}^*$ to the power of $e = (p^k - 1)/r$ is very expensive but there are some known shortcuts to make this operation more efficient. As stated in several papers (see e.g. [1]), for even embedding degrees, the above exponent can be rewritten as:

$$e = \left(p^{k/2} - 1\right)\left[\frac{p^{k/2}+1}{\Phi_k(p)}\right]\left[\frac{\Phi_k(p)}{r}\right], \quad \text{where} \quad \frac{\Phi_k(p)}{r} = \sum_{i=0}^{\varphi(k)-1} \lambda_i p^i, \qquad (4)$$

for some $\lambda_i \in \mathbb{Q}$. The first two exponentiations are easy to compute via Frobenius exponentiation. We refer therefore to the final step, raising f to the exponent $\lambda_0 + \lambda_1 p + \ldots + \lambda_{\varphi(k)-1} p^{\varphi(k)-1}$, as the "hard part" of the final exponentiation. In Section 4 we will give more details on the final exponentiation for the families that we recommend in this paper.

2.3 Security

As stated in Section 1.2, the best attack on a pairing $\widehat{e} : \mathbb{G}_2 \times \mathbb{G}_1 \to \mathbb{G}_T$ is the best attack on any of $\mathbb{G}_1$, $\mathbb{G}_2$, or $\mathbb{G}_T$. In the case of an asymmetric pairing on $E/\mathbb{F}_p$, where E has embedding degree k, the relevant groups for an attack are $\mathbb{G}_1$, which is an r-order subgroup of $E(\mathbb{F}_p)$, and $\mathbb{G}_T \subseteq \mathbb{F}_{p^k}^*$. The best known attack on $\mathbb{G}_1$ is the Pollard-rho method, which has complexity $O(\sqrt{r})$, and the best known attack on $\mathbb{G}_T$ is the number field sieve (NFS) method, or more precisely Kim and Barbulescu's special extended tower number field sieve (SexTNFS) algorithm [23], the concrete complexity of which can be computed using Barbulescu and Duquesne's method [4]. We summarize the computation of the concrete complexity in Algorithm 2. Note that the output of Algorithm 2 depends on some choices: especially on κ, h, and S. The inputs A and B can be approximated and then refined by trial and error (as described in [4]), but the choices of κ, h, and S should be approached with more care. The subtlety of choosing the best κ, h, and S is discussed in detail in [4]; we demonstrate on a case-by-case basis how to compute these inputs for our recommendations in Section 4. Finally, note that Algorithm 2 uses *Dickman's ρ-function* $\rho(v)$: this is defined to be 1 if $v \leq 1$, and is defined such that

$$d\rho/dv = -\rho(v-1)/v$$

otherwise.

The best attack complexity of computing discrete logarithms in $\mathbb{G}_T \subseteq \mathbb{F}_{p^k}^*$ previous to the attacks [23] outlined above was *guaranteed* to be worse than $\sqrt{r}$ (given that necessarily $p \geq r$) for all the embedding degrees k that were used for computing pairings. Hence the best measure at that time for finding an efficient pairing-friendly family was the *ρ-value*, given by

$$\rho = \log(p)/\log(r).$$

A family with ρ-value (close to) 1 satisfied $r \approx p$, ensuring that the $\mathbb{F}_p$-arithmetic was as efficient as possible.

Algorithm 2 Computation of the complexity of the SexTNFS Algorithm of [23], as presented in [4].

Input: A polynomial $p(x) \in \mathbb{Q}(x)$, an integer u such that $p = p(u)$ is prime, small integers κ and k such that $\kappa | k$, positive integers A and B.

Output: The cost of finding discrete logarithms in $\mathbb{F}_{p^k}$ with the SexTNFS method (for these inputs), or failure.

1: Set $\eta \leftarrow k/\kappa$.
2: Find $h \in \mathbb{Z}[t]$ such that $\deg(h) = \eta$ and h is irreducible mod p.
3: Set $\mathcal{A}$ to be the number of automorphisms of h.
4: Set w to be the number of roots of unity in $Q(t)/h(t)$.
5: Find $S(x,t) \in \mathbb{Z}[x,t]$ such that $g(x,t) \leftarrow x^\kappa + S - u$ is irreducible over $\mathbb{F}_{p^\eta} = \mathbb{F}_p[t]/(h(t))$.
6: Perform a linear change of variables on $p(x)$ to minimize the coefficients.
7: Set $f(x,t) \leftarrow p(x^\kappa + S)$.
8: Compute $\{(a_0, \ldots, a_{\eta-1}, b_0, \ldots, b_{\eta-1} \in [-A, A]^{2\eta} : a_0 \geq 0\}$.
9: Set

$$N_f \leftarrow \mathrm{Res}_t\left(\mathrm{Res}_x\left(\sum_{i=0}^{\eta-1} a_i t^i - x\sum_{i=0}^{\eta-1} b_i t^i, f(t,x)\right), h(t)\right)$$

$$N_g \leftarrow \mathrm{Res}_t\left(\mathrm{Res}_x\left(\sum_{i=0}^{\eta-1} a_i t^i - x\sum_{i=0}^{\eta-1} b_i t^i, g(t,x)\right), h(t)\right).$$

10: Set $p_f \leftarrow \rho\left(\frac{\log N_f}{\log B}\right)$ and $p_g \leftarrow \rho\left(\frac{\log N_g}{\log B}\right)$.
11: **if**

$$\frac{(2A+1)^{2\eta}}{2w} \cdot p_f \cdot p_g < \frac{2B}{\ln B}$$

then return Failure.
12: **else return**

$$\frac{2B}{\mathcal{A}\ln B} \cdot p_f^{-1} \cdot p_g^{-1} + 2^7 \cdot \frac{B^2}{\mathcal{A}\ln^2 B \cdot \log^2 B}.$$

However, in most cases asking for a ρ-value close to 1 now forces $\log(r)$ to be much larger than necessary, since $\log(p)$ has to be increased to account for the number field sieve attacks. The complexity of Algorithm 2 varies slightly for different members of the same polynomial family, due to its dependance on u. Barbulescu and Duquesne also suggested a method to compute an (best-case for the attacker) approximation for any given family: choose $h = t^\eta - t - 1$ and $S = 0$ or t. Of course we cannot check that h and g are irreducible, but this does give a first approximation for the security level for a family. The security level of a specific curve in a given family is typically 1-5 bits higher than this case.

We recommend a new additional measure of efficiency, the τ_n-*value* of a pairing-friendly family. We define the τ_n-*value* to be

$$\tau_n = \log(\sqrt{r})/2n,$$

where the family has "minimum" security level n, as computed by the method outlined above. For a family with τ_n-value (close to) 1, the complexity of the best attack on $\mathbb{G}_1$ (Pollard-rho) is approximately the same as the complexity of the best attack on $\mathbb{G}_T$ (SexTNFS).

3 New candidate families

3.1 Security level 128

Using the Brezing-Weng method [10], we generated 20 pairing-friendly families and computed, for each family, the size of $\log(p)$ and $\log(r)$ that is necessary to achieve (at least) 128-bit security, using the method of Barbulescu and Duquesne [4]. In Table 1, the value D is the CM discriminant of the elliptic curve family, and the value δ is the highest degree of twist that occurs for the family. Families 1, 13, and 17 were already presented by Fotiadis and Konstaninou in [14] together with the values of $\log(p)$ and $\log(r)$ corresponding to a approximate security level computation via the asymptotic formula $L_{p^k}[1/3, 1.529]$.

Comparing the families in Table 1, Family 17 is a clear winner (at least given only this information). It has the most efficient $\mathbb{F}_p$- and $\mathbb{F}_{p^{k/\delta}}$-arithmetic; the efficient $\mathbb{F}_p$-arithmetic is a consequence of the fact that Family 17 has both the smallest τ_{128}-value and the smallest ρ-value of the table. Also, the degrees of both defining polynomials p and r are the smallest of the table, and having low degree polynomials contributes to a more efficient final exponentiation. Recall from Algorithm 1 that the efficiency of the optimal ate pairing relies also on the short vector of the lattice. For Family 17 we choose the short vector:

$$[6x + 2, -1, -1, -1],$$

which contains only one non-constant term, and this is (only) linear.

For all of the above reasons, we assume that the most efficient family of Table 1 with respect to pairing computation is Family 17. In Section 4, we give more details on the efficiency of computing a pairing on two specific elliptic curves in this family and show that this family produces more efficient choices compared with other recommendations in the literature.

For the full information on each family in Table 1 (defining polynomials, short vectors) please see Appendix C.

Table 1. Candidate families for Security Level 128

Label	k	D	$\deg(r)$	$\deg(p)$	$\log(p)$	$k\log(p)$	$k/\delta\log(p)$	ρ	τ_{128}
1	8	1	4	8	760	6080	1520	2	1.48
2	8	1	4	8	760	6080	1520	2	1.48
3	8	2	4	8	768	6144	3072	2	1.5
4	8	3	8	16	512	4906	2048	2	1
5	8	1	4	8	752	6016	1504	2	1.47
6	8	1	4	8	704	5632	1408	2	1.375
7	8	1	4	8	752	6016	1504	2	1.47
8	8	1	4	8	752	6016	1504	2	1.47
9	8	1	8	16	512	4096	1024	2	1
10	9	3	6	12	624	5616	1872	2	1.22
11	9	3	6	12	516	4644	1548	2	1.008
12	9	3	6	12	512	4608	1536	2	1
13	10	1	8	14	448	4480	2240	1.75	1
14	10	5	8	14	448	4480	2240	1.75	1
15	10	15	8	14	448	4480	2240	1.75	1
16	10	1	8	14	448	4480	2240	1.75	1
17	12	3	4	6	384	4608	768	1.5	1
18	12	2	8	14	448	5376	2688	1.75	1
19	12	3	4	6	444	5328	888	1.5	1.16
20	12	3	4	6	480	5760	960	1.5	1.25

3.2 Security level 192

Using the Brezing-Weng method [10], we generated 7 pairing-friendly families and, for each family, computed the size of $\log(p)$ and $\log(r)$ that is necessary to achieve (at least) 192-bit security via the method of Barbulescu and Duquesne [4].

Table 2. Candidate families for Security Level 192

Label	k	D	$\deg(r)$	$\deg(p)$	$\log(p)$	$k\log(p)$	$k/\delta\log(p)$	ρ	τ_{192}
21	15	3	8	16	784	11760	3920	2	1.02
22	15	3	8	16	768	11520	3840	2	1
23	16	1	8	16	768	12288	3072	2	1
24	16	1	8	16	768	12288	3072	2	1
25	18	3	6	12	792	14256	2376	2	1.03
26	18	3	6	12	768	13824	2304	2	1
27	20	1	8	12	648	12960	3240	1.5	1.125

Families 21, 23, and 27 were already presented by Fotiadis and Konstaninou in [14] together with the values of $\log(p)$ and $\log(r)$ corresponding to a security level computation with the asymptotic formula $L_{p^k}[1/3, 1.529]$.

Where for security level 128, the comparison table gave a clear answer, for security level 192 there are many similar options. We chose to analyze families 23 and 25 in more detail as both of these families have simple defining polynomials $p(x)$ and $r(x)$, which helps to reduce the cost of the final exponentiation; both families also have a good choice of short vector (although this is true for more families). For the full information on each family of Table 2 (eg. defining polynomials, short vectors) please see Appendix C.

4 Recommendations

As justified in Section 3, for security level 128, we recommend Family 17 from Table 1:

Family 17: $k = 12$, $D = 3$, $\rho = 1.5$, $\tau_{128} = 1$, sextic twists.

Defining polynomials:

$$p(x) = 1728x^6 + 2160x^5 + 1548x^4 + 756x^3 + 240x^2 + 54x + 7.$$
$$t(x) = -6x^2 + 1, \quad r(x) = 36x^4 + 36x^3 + 18x^2 + 6x + 1.$$

- Short vector: $[6x + 2, -1, -1, -1]$.
- **Choice (a)**: $u = -2^{64} - 2^{63} - 2^{11} - 2^{10}$, NAF-hw$(6u + 2) = 5$, $\log(p(u)) = 398$, $\log(r(u)) = 257$.
- **Choice (b)**: $u = -2^{72} - 2^{71} - 2^{36}$, NAF-hw$(6u + 2) = 5$, $\log(p(u)) = 446$, $\log(r(u)) = 296$.

We want the NAF-hamming weight of both u and $6u + 2$ to be minimal in this case, for minimizing the cost of the final exponentiation and the Miller loop respectively. It is of course possible to work directly with $6u + 2$ in the final exponentiation, but this yielded less efficient results in our case.

The security levels for choices (a) and (b) are 127 and 131 respectively. The motivation for these two choices is as follows: choice (a) has the smallest $\log(p)$ among the (almost) 128-bit secure choices with low hamming weight. Choice (a) is also twist-secure, but not subgroup secure. However, subgroup attacks can be avoided using membership checks [6]. Choice (b) has the highest $\log(p)$ among the choices with NAF-hw$(u) \leq 3$ that can still be written in seven 64-bit words, so could be a good choice for 64-bit architecture. It is however *not* twist secure.

4.1 Computing the cost of finite field arithmetic

To compute the cost of the finite field arithmetic, we follow Guillevic, Masson, and Thomé [20].

Notation

$\mathbf{m}$	Multiplication in $\mathbb{F}_p$
$\mathbf{m}_i$	Multiplication in $\mathbb{F}_{p^i}$
$\mathbf{s}_i$	Squaring in $\mathbb{F}_{p^i}$
$\mathbf{i}_i$	Inversion in $\mathbb{F}_{p^i}$
$\mathbf{f}_i$	Exponentiation by p^i in $\mathbb{F}_{p^k}$
$\mathbf{e}_u$	Exponentiation by u in $\mathbb{F}_{p^k}$
$\mathbf{s}_i^{\text{cyclo}}$	Cyclotomic squaring in $\mathbb{F}_{p^i}$

Table 3 gives the cost, in terms of $\mathbf{m}$, of all the extension field arithmetic operations that occur in the computation of the pairing for Family 17. The costs in this case are from [20, Table 5]. This is assuming that Karatsuba-style methods are followed for multiplication, and assuming the generic formula that $\mathbf{f}_i = (k-2)\mathbf{m}$ if k is even.

We use the generic formula for $\mathbf{f}_i$ here because it is the fastest method in these instances. The other natural method supposes that $i|k$: Define ω such that $\mathbb{F}_{p^k} = \mathbb{F}_{p^i}[\omega]$. Then for $a \in \mathbb{F}_{p^k}$ we can compute a^{p^i} via

$$\begin{aligned} a^{p^i} &= (a_0 + a_1\omega + \cdots + a_{k/i-1}\omega^{k/i-1})^{p^i} \\ &= a_0 + a_1\omega^{p^i} + \cdots + a_{k/i-1}\omega^{(k/i-1)p^i}, \end{aligned}$$

which as the powers of ω can be precomputed, costs $(k/i-1)\mathbf{m}_i$.

Table 3. Costs of extension field arithmetic when $k = 12$

$\mathbf{m}_2$	$\mathbf{m}_{12}$	$\mathbf{s}_1$	$\mathbf{s}_2$	$\mathbf{s}_{12}$	$\mathbf{i}_1$	$\mathbf{i}_2$	$\mathbf{i}_{12}$	$\mathbf{f}_i$, $i \neq 6, 12$
$3\mathbf{m}$	$54\mathbf{m}$	$\mathbf{m}$	$2\mathbf{m}$	$36\mathbf{m}$	$25\mathbf{m}$	$4\mathbf{m} + \mathbf{i}_1$	$94\mathbf{m} + \mathbf{i}_1$	$10\mathbf{m}$

4.2 Computing the cost of the Miller loop

Plugging in the values for Family 17 into (3), we see that optimal ate pairing of $(Q, P) \in \mathbb{G}_2 \times \mathbb{G}_1$ is given by

$$\widehat{a}(Q, P) = (f_{6u+2,Q}(P) \cdot H)^{\frac{p^k-1}{r}}.$$

From Appendix B we see that the most efficient computation of the optimal ate pairing in the case of degree 6 twists is from [11, Section 5]: with $k = 12$ each addition step (cf. Steps 7, 8, and 9 of Algorithm 1) costs a total of

$$23\mathbf{m}_2 + 2\mathbf{s}_2 + 4\mathbf{m} = 77\mathbf{m}$$

and each doubling step (cf. Steps 4 and 5 of Algorithm 1) costs a total of

$$15\mathbf{m}_2 + 7\mathbf{s}_2 + \mathbf{s}_{12} + 4\mathbf{m} = 99\mathbf{m},$$

except for the first, which costs

$$2\mathbf{m}_2 + 7\mathbf{s}_2 + 4\mathbf{m} = 24\mathbf{m}.$$

For choice (a), our short vector $6u + 2$ has NAF-hamming weight 5 and the binary expansion of $6u + 2$ is length 68, so the computation of $f_{6u+2,Q}(P)$ costs

$$24\mathbf{m} + 66 \cdot 99\mathbf{m} + 4 \cdot 77\mathbf{m} = 6866\mathbf{m}.$$

Similarly, for choice (b), the computation of $f_{6u+2,Q}(P)$ costs $7218\mathbf{m}$.

Finally, we must compute the factor H, where

$$H = h_{-(p+p^2+p^3)Q,xQ}(P) \cdot h_{-(p^2+p^3)Q,-pQ}(P) \cdot h_{-p^3Q,-p^2Q}(P).$$

Recall that the group order r divides $6u + 2 - p - p^2 - p^3$, so

$$-(p + p^2 + p^3)Q = -(6u + 2)Q, \tag{5}$$

hence $h_{-(p+p^2+p^3)Q,(6u+2)Q}(P)$ goes to 1 in the final exponentiation. The denominators of $h_{-(p^2+p^3)Q,-pQ}(P)$ and $h_{-p^3Q,-p^2Q}(P)$ go to 1 in the final exponentiation by construction of Q, so by (5) we can put

$$h_{-(p^2+p^3)Q,-pQ}(P) = \ell_{(6u+2)Q,-pQ}(P)$$

and

$$h_{-p^3Q,-p^2Q}(P) = \ell_{((6u+2)-p)Q,-p^2Q}(P).$$

Therefore, the total cost for computing H is the cost of computing pQ and p^2Q, one point addition (to compute $pQ - (6u + 2)Q$), two evaluations at P, and one multiplication.

- As $Q = [X_Q, Y_Q, Z_Q] \in \mathbb{G}_2 \supseteq \ker([p]-\pi)$, we can compute pQ as $[X_Q^p, Y_Q^p, Z_Q^p]$. Furthermore, by construction of Q we have that $Z_Q \in \mathbb{F}_{p^2}$, so Z_Q^p amounts to conjugation of Z_Q in $\mathbb{F}_{p^2}/\mathbb{F}_p$. In other words, the computation of pQ costs $2\mathbf{f}_1 = 20\mathbf{m}$, and similarly the computation of $p^2Q = p \cdot pQ$ costs another $2\mathbf{f}_1 = 20\mathbf{m}$.
- The points being added are images under the degree six twist isomorphism of points with coordinates in $\mathbb{F}_{p^2}$, so the cost is for point addition in $\mathbb{F}_{p^2}$. Using the point addition formulae from [7], this gives $11\mathbf{m}_2 + 5\mathbf{s}_2 = 43\mathbf{m}$.
- For both line functions, the gradient is an element of $\omega\mathbb{F}_{p^2}$, where $\mathbb{F}_{p^{12}} = \mathbb{F}_{p^2}(\omega)$, by construction of Q. Therefore the evaluation of each line at P can be performed by multiplying an element of $\mathbb{F}_p$ with an element of $\mathbb{F}_{p^2}$, which costs $2\mathbf{m}$, giving a total cost of $4\mathbf{m}$ for evaluation.
- The multiplication of the two line functions to obtain H is a multiplication in $\mathbb{F}_{p^{12}}$, costing $54\mathbf{m}$.

For Family 17a the cost of the Miller loop is therefore the cost $6866\mathbf{m}$ of computing $f_{6u+2,Q}(P)$ via Miller's algorithm, plus $141\mathbf{m}$ for the computation of H, plus $54\mathbf{m}$ for multiplying by H. All together, this gives a total cost of $7061\mathbf{m}$ for the whole Miller loop. Similarly, for Family 17b, the total cost of the Miller loop is $7853\mathbf{m}$.

4.3 Final exponentiation

Recall that the exponent $e = (p^{12}-1)/r$ can be equivalently written in the form

$$e = (p^6-1)(p^2+1)\frac{\Phi_{12}(p)}{r}.$$

The "easy" part $f \leftarrow f^{(p^6-1)(p^2+1)}$ can be computed with $2\mathbf{f}_i + 1\mathbf{m}_{12} + 1\mathbf{i}_{12}$ operations. We follow [17] to efficiently compute the "hard" part. Using their techniques, we find that

$$u(12u^2+6u+1)\frac{\Phi_{12}(p(u))}{r(u)} = \lambda_3(u)p(u)^3 + \lambda_2(u)p(u)^2 + \lambda_1(u)p(u) + \lambda_0(u),$$

where the polynomials λ_i are defined as:

$$\begin{aligned}
\lambda_3 &= -288u_0^5 + 360u_0^4 - 162u_0^3 + 54u_0^2 - 6u_0\\
\lambda_2 &= -288u_0^5 + 360u_0^4 - 210u_0^3 + 66u_0^2 - 13u_0\\
\lambda_1 &= -288u_0^5 + 360u_0^4 - 306u_0^3 + 138u_0^2 - 39u_0 + 7\\
\lambda_0 &= 576u_0^5 - 432u_0^4 + 204u_0^3 - 54u_0^2 + 5u_0 - 1,
\end{aligned}$$

with $u_0 = -u$.

The details of our computation of the final exponentiation is given in Appendix A. The bottleneck of the computation is the five exponentiations by u_0, each of which costs

$$1\mathbf{e}_{u_0} = 4(\log(u_0)-1)\mathbf{m}_2 + (6N-3)\mathbf{m}_2 + N\mathbf{m}_{12} + 3N\mathbf{s}_2 + 1\mathbf{i}_2 = 1022\mathbf{m},$$

where $N = \text{NAF-hw}(u_0) - 1 = 3$, see [22]. The total cost for computing the final exponentiation is:

$$5\mathbf{e}_{u_0} + 8\mathbf{f}_i + 18\mathbf{s}_{12}^{\text{cyclo}} + 40\mathbf{m}_{12} + 2\mathbf{i}_{12} = 7912\mathbf{m},$$

where by [20], $1\mathbf{s}_{12}^{\text{cyclo}} = 18\mathbf{m}$.

We do not claim that this is the best implementation of the final exponentiation– further improvements may be possible.

4.4 Computing the security level

We also searched for the best choices of κ, h, and S, in Algorithm 2 to give the most effective attack, in order to more carefully analyze the security. By [4, Table 5], the best choice of κ for $k = 12$ is either 1 or 2. The change of variables that we apply for Step 6 of Algorithm 2 is $p(x) \leftarrow 108 \cdot p((x-6)/2)$, which results in

$$p(x) = 4x^6 - 18x^5 + 69x^4 - 94x^3 + 108x^2 + 132x + 28;$$

thank you to Aurore Guillevic for this suggestion.

Choice (a) With $\kappa = 2$, we get $\eta = 6$ as $k = \kappa \cdot \eta$, so we search for an irreducible polynomial $h(t) \in \mathbb{F}_p[t]$ of degree 6. For the best attack we would like $h(t)$ to have many automorphisms (more than 2)–the only choices for such an h (of small level) are the cyclotomic polynomials Φ_7 and Φ_9 [4, Table 4] which in this case are both reducible. So we search manually for an irreducible polynomial $h(t)$ with small coefficients (again to make the attack as effective as possible). In this case

$$h(t) = t^6 - t^4 + t^2 + 1$$

is an irreducible polynomial (and the best one we could find). Finally, we search for $S(t, x)$ as small as possible such that $g(t, x) = x^2 + S - u$ is irreducible over $\mathbb{F}_p[t]/h(t)$. In this case, we found that $S(t, x) = t$ satisfies this. The SexTNFS algorithm performed with these choices has complexity approximately $2^{127.4}$.

With $\kappa = 1$, we get $\eta = 12$ as $k = \kappa \cdot \eta$, so we search for an irreducible polynomial $h(t) \in \mathbb{F}_p[t]$ of degree 12. For the best attack we would like $h(t)$ to have many automorphisms so again we check for cyclotomic polynomials. In this case the only choice (of small level) is $h(t) = \Phi_{13}(t)$, which is reducible. (In fact, this u was chosen so that Φ_7, Φ_9, and Φ_{13} are all reducible). Searching manually for an irreducible polynomial $h(t)$, the most effective choice we find for the attack is

$$h(t) = t^{12} - t^4 - t^3 + t^2 - 1.$$

In this case, the smallest $S(t, x)$ we found such that $g = x^2 + S - u$ is irreducible over $\mathbb{F}_p[t]/h(t)$ was $S = t^2$. The SexTNFS algorithm performed with these choices has complexity approximately $2^{129.5}$.

In conclusion, the most effective parameters for the SexTNFS algorithm that we found were $\kappa = 2$, $h = t^6 - t^4 + t^2 + 1$, and $S = t$, giving a complexity of $2^{127.4}$.

Choice (b) The calculation follows the same procedure as for choice (a). The most effective parameters for the SexTNFS algorithm that we found in this case were $\kappa = 2$, $h = t^6 + t - 1$, and $S = t + 1$, giving a complexity of $2^{130.5}$. The best choices for h and S that we found with $\kappa = 1$ were $h = \Phi_{13}(t)$ and $S = t + 1$, giving a complexity of $2^{133.4}$.

4.5 Comparison with the literature

Guillevic, Masson, and Thomé [20, Table 9] presented a summary of the most efficient known pairings providing 128-bit security: BN, BLS12, KSS16, and their new families.

Table 4. $\mathbb{F}_p$-multiplication timings as given in [20]

$\log(p)$	$[324, 384]$	$[385, 447]$	$[448, 512]$	$[512, 576]$	$[640, 704]$
m	69ns	≈90ns*	120ns	154ns	230ns

* Not given in [20]. Estimation based on clock cycles (for easy comparison).

Table 5. Number of clock cycles per $\mathbb{F}_p$-multiplication using generic Montgomery-schoolbook methods

$\log(p)$	$[324, 384]$	$[385, 447]$	$[448, 512]$	$[512, 576]$	$[640, 704]$
m	99	129	166	210^{**}	314^{**}

** This is an estimate; we did not implement these cases.

We compare our Family 17 with these examples in Table 6. We refer to the timings given in [20, Table 8] for comparison only; those timings are also written here in Table 4 for the convenience of the reader. Lorenz Panny was kind enough to compute for us the number of clock cycles, using generic Montgomery-schoolbook methods for multiplication in $\mathbb{F}_p$, for 64-bit words p of length 6, 7, or 8; see Table 5.

Table 6. Comparison of Family 17 with all known examples of $\leq 3 \times 10^6$ clock cycles

Curve	$\log(p)$	Miller loop	Final exponentiation	Time	Clock cycles
BN	462	12180**m**	5691**m**	≈2.20ms	2966586
$k = 6$ [20]	672	4601**m**	3871**m**	≈1.95ms	≈2660208
KSS16	339	7691**m**	18235**m**	≈1.79ms	2566674
$k = 8$ [20]	544	4502**m**	7134**m**	≈1.79ms	≈2443560
BLS12	461	7685**m**	6193**m**	≈1.67ms	2303748
Family 17 (a)	398	7061**m**	7912**m**	≈1.35ms	1931517
Family 17 (b)	446	7853**m**	8002**m**	≈1.43ms	2045295

Remark Very recently, Barbulescu, El Mrabet, and Ghammam presented more candidate families [5], which they claim to be "very competitive with BN, BLS12 and KSS16 at 128 bits of security". However, we leave a careful analysis of the number of multiplications in the Miller loop and the final exponentiation for a good member of each of their families, in order for a full comparison, for future work.

4.6 Two recommendations for security level 192

As justified in Section 3, we recommend Families 23 and 25 from Table 2.

Family 23

$k = 16$, $D = 1$, $\rho = 2$, $\tau_{192} = 1$, quartic twists.

Defining polynomials:

$$p(x) = (x^{16} + x^{10} + 5x^8 + x^2 + 4x + 4)/4, t(x) = x^8 + x + 2, r(x) = x^8 + 1.$$

- Short vector: $[x, -1, 0, 0, 0, 0, 0, 0]$.
- Gives prime values for $p(u)$ and $r(u)$ when $u = 2^{48} + 2^{28} + 2^{26}$.
- Twist-secure for this u.
- $\log(r) = 384$ and $\log(p) = 766$.

The formula for the optimal ate pairing in this example is:

$$\widehat{a}(Q, P) = f_{u,Q}(P)^{\frac{p^{16}-1}{r}}.$$

Table 7 shows the costs of the extension field arithmetic occurring in the computation of a pairing on an elliptic curve in Family 23. Again, we follow [20] to compute the cost of each operation in terms of $\mathbb{F}_p$-multiplications $\mathbf{m}$. The costs for $\mathbf{m}_{16}$, $\mathbf{s}_{16}$, $\mathbf{i}_{16}$, and $\mathbf{f}_i$ are in [20, Table 5]. We computed $\mathbf{m}_4$ and $\mathbf{s}_4$ via $\mathbf{m}_4 = \mathbf{m}_2^2$ and $\mathbf{s}_4 = 2 \cdot \mathbf{s}_2 + \mathbf{m}_2$.

Table 7. Costs of extension field arithmetic when $k = 16$

$\mathbf{m}_4$	$\mathbf{m}_{16}$	$\mathbf{s}_1$	$\mathbf{s}_4$	$\mathbf{s}_{16}$	$\mathbf{i}_{16}$	$\mathbf{f}_i$, $i \neq 8, 16$
$9\mathbf{m}$	$81\mathbf{m}$	$\mathbf{m}$	$7\mathbf{m}$	$54\mathbf{m}$	$134\mathbf{m} + \mathbf{i}_1$	$14\mathbf{m}$

For Family 23, the elliptic curve $E/\mathbb{F}_p$ on which we compute the pairing has quartic twists, which we make use of to perform extension field arithmetic in $\mathbb{F}_{p^4}$ instead of $\mathbb{F}_{p^{16}}$– this requires performing elliptic curve arithmetic on the quartic twist $E^t/\mathbb{F}_{p^4}$ of E. There are two known curve forms for which both E and E^t can be written in the same form: Weierstrass and Jacobi Quartic. Writing E and E^t in different forms would be expensive; we would pay for many costly curve conversions. Comparing the fastest Miller addition and Miller doubling formulas on Weierstrass curves and Jacobi Quartic curves in Appendix B we find that Weierstrass curves are the more efficient choice. With the Weierstrass formulae (see Appendix B for more details), we obtain a cost of

$$47(10\mathbf{m}_4+8\mathbf{s}_4+8\mathbf{m}+\mathbf{s}_{16})+2(17\mathbf{m}_4+5\mathbf{s}_4+8\mathbf{m})+(2\mathbf{m}_4+8\mathbf{s}_4+8\mathbf{m})+\mathbf{m}_{16} = 10331\mathbf{m}$$

for computing the Miller loop.

By Eq. (4), the final exponentiation in this case is equivalently written as:

$$e = (p^8 + 1)(\lambda_7 p^7 + \lambda_6 p^6 + \lambda_5 p^5 + \lambda_4 p^4 + \lambda_3 p^3 + \lambda_2 p^2 + \lambda_1 p + \lambda_0),$$

where $\lambda_7 = (u^8 + u^2 + 4)/4$. The remaining λ_i are calculated recursively by $\lambda_i = u\lambda_{i+1}$, for $i = 1, 2, 3, 4, 5, 6$ and $\lambda_0 = u\lambda_1 + 1$. Then the hard part is:

$$f^{\frac{p^8+1}{r}} = y^{\sum_{i=0}^{7}(u^i p^{7-i})} f \quad \text{with} \quad y = f^{\frac{u^8+u^2+4}{4}} = \left[\left(f^{\frac{u}{2}}\right)^{\frac{u}{2}}\right]^{u^6} \left(f^{\frac{u}{2}}\right)^{\frac{u}{2}} f,$$

where by [20], since $1\mathbf{s}_{16}^{\text{cyclo}} = 36\mathbf{m}$, the exponentiations by $u/2$ and u cost:

$$1\mathbf{e}_{\frac{u}{2}} = (\log(u/2) - 1)\mathbf{s}_{16}^{\text{cyclo}} + (\text{wt}(u/2) - 1)\mathbf{m}_{16} = 1818\mathbf{m}$$
$$1\mathbf{e}_u = (\log(u) - 1)\mathbf{s}_{16}^{\text{cyclo}} + (\text{wt}(u) - 1)\mathbf{m}_{16} = 1854\mathbf{m}$$

In conclusion, the total cost for the final exponentiation in this case is (see also Appendix A):

$$8\mathbf{f}_i + 2\mathbf{e}_{\frac{u}{2}} + 13\mathbf{e}_u + 12\mathbf{m}_{16} + 1\mathbf{i}_{16} = 28981\mathbf{m}.$$

Hence the total cost for calculating the pairing is: $39312\mathbf{m}$.

Finally, the most effective parameters for the SexTNFS algorithm that we found were $\kappa = 1$, $h = \Phi_{17}(t)$, and $S = t + 1$, giving a complexity of $2^{196.4}$.

Family 25

$k = 18$, $D = 3$, $\rho = 2$, $\tau_{192} = 1.03$, sextic twists.

Defining polynomials:

$$p(x) = (3x^{12} - 3x^9 + x^8 - 2x^7 + 7x^6 - x^5 - x^4 - 4x^3 + x^2 - 2x + 4)/3.$$
$$t(x) = x^6 - x^4 - x^3 + 2, \quad r(x) = x^6 - x^3 + 1.$$

- Short vector: $[x, 1, 0, 0, -1, 0]$.
- Gives prime values for $p(u)$ and $r(u)$ when $u = -2^{64} - 2^{35} + 2^{11} - 1$.
- $\log(r) = 384$ and $\log(p) = 768$.

The formula of the optimal ate pairing in this case is given by:

$$\widehat{a}(Q, P) = f_{u,Q}(P)^{\frac{p^{18}-1}{r}}.$$

One may think at first sight that, as $\tau_{128} > 1$, choosing a value of u for which $\log(r(u)) = 384$ would not give the required the security level. This is a valid concern, but recall that the computation of the security level before choosing a parameter u is typically an underestimation by between 1 and 5 bits; in this case in turns out to be an underestimation by about 5 bits, so this choice of u is sufficient.

Table 8 shows the costs of the extension field arithmetic occurring in the computation of a pairing on an elliptic curve in Family 25. Again, we follow [20] to compute the cost of each operation in terms of $\mathbb{F}_p$-multiplications $\mathbf{m}$. The

costs for $\mathbf{m}_3$ and $\mathbf{s}_3$ are also in [20, Table 5]. We computed $\mathbf{i}_{18}$ via the recursive formulae $\mathbf{i}_{3i} = \mathbf{i}_i + 3\mathbf{s}_i + 9\mathbf{m}_i$ and $\mathbf{i}_{2i} = \mathbf{i}_i + 2\mathbf{s}_i + 2\mathbf{m}_i$ as is done in [20] with other examples. We computed $\mathbf{m}_{18}$ via $\mathbf{m}_{18} = 6 \cdot \mathbf{m}_6 = 6 \cdot (6 \cdot \mathbf{m}_2)$, that is, using Karatsuba multiplication for cubic field extensions as outlined in [12]. The cost for $\mathbf{s}_{18}$ is from [18].

Table 8. Costs of extension field arithmetic when $k = 18$

$\mathbf{m}_3$	$\mathbf{m}_{18}$	$\mathbf{s}_1$	$\mathbf{s}_3$	$\mathbf{s}_{18}$	$\mathbf{i}_{18}$	$\mathbf{f}_i$, $i \neq 9, 18$
$6\mathbf{m}$	$108\mathbf{m}$	$\mathbf{m}$	$5\mathbf{m}$	$72\mathbf{m}$	$208\mathbf{m} + \mathbf{i}_1$	$16\mathbf{m}$

For Family 25, the elliptic curve $E/\mathbb{F}_p$ on which we compute the pairing has sextic twists, which we make use of to perform extension field arithmetic in $\mathbb{F}_{p^3}$ instead of $\mathbb{F}_{p^{18}}$– this requires performing elliptic curve arithmetic on the sextic twist $E^t/\mathbb{F}_{p^3}$ of E. The only known curve form for which both E and E^t can be written in the same form is Weierstrass.

With the Weierstrass formulae (see Appendix B for more details), we obtain a cost of

$$43(15\mathbf{m}_3+7\mathbf{s}_3+6\mathbf{m}+\mathbf{s}_{18})+3(23\mathbf{m}_3+2\mathbf{s}_3+6\mathbf{m})+(2\mathbf{m}_3+7\mathbf{s}_3+6\mathbf{m})+\mathbf{m}_{18} = 13412\mathbf{m}$$

for computing the Miller loop.

For the final exponentiation, by Eq. (4) we get $e = (p^9-1)(p^3+1)(\Phi_{18}(p)/r)$. Again following the techniques of [17], we write $\Phi_{18}(p)/r$ as $\sum_{i=0}^{5} \lambda_i p^i$, where:

$$\begin{aligned}
\lambda_5 &= 3u_0^{10} + u_0^6 + 2u_0^5 + 4u_0^4 - 3u_0^2 \\
\lambda_4 &= 3u_0^{11} + 3u_0^8 + u_0^7 + 2u_0^6 + 4u_0^5 + u_0^4 - u_0^3 + 4u_0^2 \\
\lambda_3 &= 3u_0^9 + u_0^5 + 2u_0^4 + 4u_0^3 - 3u_0 \\
\lambda_2 &= 3u_0^7 + u_0^3 + 2u_0^2 + 4u_0 \\
\lambda_1 &= -(3u_0^{11} + u_0^7 + 2u_0^6 + 4u_0^5 - 3u_0^3 + 3) \\
\lambda_0 &= 3u_0^6 + u_0^2 + 2u_0 + 4
\end{aligned}$$

with $u = -u_0$ and $u_0 = 2^{64} + 2^{35} - 2^{11} + 1$ (see also Appendix A). The total cost for computing the final exponentiation is:

$$11\mathbf{e}_u + 7\mathbf{f}_i + 4\mathbf{s}_{18}^{\text{cyclo}} + 15\mathbf{m}_{18} + 4\mathbf{i}_{18} = 24896\mathbf{m},$$

where $1\mathbf{s}_{18}^{\text{cyclo}} = 36\mathbf{m}$.

Finally, the most effective parameters for the SexTNFS algorithm that we found were $\kappa = 1$, $h = t^{18} - t^5 + 1$, and $S = t$, giving a complexity of $2^{194.2}$.

Remark. In [1], Aranha, Fuentes-Castañeda, Knapp, Menezes, and Rodríguez-Henríquez studied the implementation of pairings that aim at a security level of 192-bits. However this paper appeared before the Kim and Barbulescu attack [23], so we do not compare our examples with [1]. Again, in future work

we would like to do a comparison with the new work of Barbulescu, El Mrabet, and Loubna Ghammam [5], but this requires doing precise computations for a well-chosen member of each family that they propose, which is beyond the scope of this paper.

A Final exponentiation algorithms

We include here the algorithms that we used to attempt to optimize the final exponentiation step for our three recommended families. We would welcome any method for computing the final exponentiation that improves upon these algorithms.

Algorithm 3 Final exponentiation for Family 17 with $u < 0$.

Input: Some $\mathtt{u} \in \mathbb{Z}_{<0}$ such that $\mathtt{p} = \mathtt{p}(\mathtt{u})$ and $\mathtt{r} = \mathtt{r}(\mathtt{u})$ are prime; an element $\mathtt{f} \in \mathbb{F}_{\mathtt{p}^{12}}$

Output: The value $\mathtt{f}^{\frac{\mathtt{p}^{12}-1}{\mathtt{r}}}$

$\mathtt{u} \leftarrow |\mathtt{u}|$
$\mathtt{t} \leftarrow \mathtt{f}^{\mathtt{p}^6}; \mathtt{f} \leftarrow \mathtt{t}/\mathtt{f}; \mathtt{f} \leftarrow \mathtt{f}^{\mathtt{p}^2}\mathtt{f};$
//compute powers of f
$\mathtt{f}_2 \leftarrow \mathtt{f}^2; \mathtt{f}_3 \leftarrow \mathtt{f}_2\mathtt{f} \leftarrow \mathtt{f}^3; \mathtt{f}_6 \leftarrow \mathtt{f}_3^2; \mathtt{f}_7 \leftarrow \mathtt{f}_6\mathtt{f};$
//compute powers of $\mathtt{f}^{\mathtt{u}}$
$\mathtt{f}_{\mathtt{u}1} \leftarrow \mathtt{f}^{\mathtt{u}}; \mathtt{f}_{\mathtt{u}2} \leftarrow \mathtt{f}_{\mathtt{u}1}^2; \mathtt{f}_{\mathtt{u}4} \leftarrow \mathtt{f}_{\mathtt{u}2}^2; \mathtt{f}_{\mathtt{u}5} \leftarrow \mathtt{f}_{\mathtt{u}4}\mathtt{f}_{\mathtt{u}1}; \mathtt{f}_{\mathtt{u}6} \leftarrow \mathtt{f}_{\mathtt{u}5}\mathtt{f}_{\mathtt{u}1}; \mathtt{f}_{\mathtt{u}8} \leftarrow \mathtt{f}_{\mathtt{u}4}^2;$
$\mathtt{f}_{\mathtt{u}13} \leftarrow \mathtt{f}_{\mathtt{u}8}\mathtt{f}_{\mathtt{u}5}; \mathtt{f}_{\mathtt{u}26} \leftarrow \mathtt{f}_{\mathtt{u}13}^2; \mathtt{f}_{\mathtt{u}39} \leftarrow \mathtt{f}_{\mathtt{u}26}\mathtt{f}_{\mathtt{u}13};$
//compute powers of $\mathtt{f}^{\mathtt{u}^2}$
$\mathtt{f}_{2\mathtt{u}6} \leftarrow \mathtt{f}_{\mathtt{u}6}^{\mathtt{u}}; \mathtt{f}_{2\mathtt{u}12} \leftarrow \mathtt{f}_{2\mathtt{u}6}^2; \mathtt{f}_{2\mathtt{u}24} \leftarrow \mathtt{f}_{2\mathtt{u}12}^2; \mathtt{f}_{2\mathtt{u}48} \leftarrow \mathtt{f}_{2\mathtt{u}24}^2; \mathtt{f}_{2\mathtt{u}54} \leftarrow \mathtt{f}_{2\mathtt{u}48}\mathtt{f}_{2\mathtt{u}6};$
$\mathtt{f}_{2\mathtt{u}66} \leftarrow \mathtt{f}_{2\mathtt{u}54}\mathtt{f}_{2\mathtt{u}12}; \mathtt{f}_{2\mathtt{u}132} \leftarrow \mathtt{f}_{2\mathtt{u}66}^2; \mathtt{f}_{2\mathtt{u}138} \leftarrow \mathtt{f}_{2\mathtt{u}132}\mathtt{f}_{2\mathtt{u}6};$
//compute powers of $\mathtt{f}^{\mathtt{u}^3}$
$\mathtt{f}_{3\mathtt{u}6} \leftarrow \mathtt{f}_{2\mathtt{u}6}^{\mathtt{u}}; \mathtt{f}_{3\mathtt{u}12} = \mathtt{f}_{3\mathtt{u}6}^2; \mathtt{f}_{3\mathtt{u}18} \leftarrow \mathtt{f}_{3\mathtt{u}12}\mathtt{f}_{3\mathtt{u}6}; \mathtt{f}_{3\mathtt{u}36} \leftarrow \mathtt{f}_{3\mathtt{u}18}^2; \mathtt{f}_{3\mathtt{u}72} \leftarrow \mathtt{f}_{3\mathtt{u}36}^2;$
$\mathtt{f}_{3\mathtt{u}144} \leftarrow \mathtt{f}_{3\mathtt{u}72}^2; \mathtt{f}_{3\mathtt{u}162} \leftarrow \mathtt{f}_{3\mathtt{u}144}\mathtt{f}_{3\mathtt{u}18}; \mathtt{f}_{3\mathtt{u}204} \leftarrow \mathtt{f}_{3\mathtt{u}162}\mathtt{f}_{3\mathtt{u}36}\mathtt{f}_{3\mathtt{u}6};$
$\mathtt{f}_{3\mathtt{u}210} \leftarrow \mathtt{f}_{3\mathtt{u}204}\mathtt{f}_{3\mathtt{u}6}; \mathtt{f}_{3\mathtt{u}288} \leftarrow \mathtt{f}_{3\mathtt{u}144}^2; \mathtt{f}_{3\mathtt{u}306} \leftarrow \mathtt{f}_{3\mathtt{u}288}\mathtt{f}_{3\mathtt{u}18};$
//compute powers of $\mathtt{f}^{\mathtt{u}^4}$
$\mathtt{f}_{4\mathtt{u}72} \leftarrow \mathtt{f}_{3\mathtt{u}72}^{\mathtt{u}}; \mathtt{f}_{4\mathtt{u}144} \leftarrow \mathtt{f}_{4\mathtt{u}72}^2; \mathtt{f}_{4\mathtt{u}288} \leftarrow \mathtt{f}_{4\mathtt{u}144}^2; \mathtt{f}_{4\mathtt{u}360} \leftarrow \mathtt{f}_{4\mathtt{u}288}\mathtt{f}_{4\mathtt{u}72};$
$\mathtt{f}_{4\mathtt{u}432} \leftarrow \mathtt{f}_{4\mathtt{u}360}\mathtt{f}_{4\mathtt{u}72};$
//compute powers of $\mathtt{f}^{\mathtt{u}^5}$
$\mathtt{f}_{5\mathtt{u}288} \leftarrow \mathtt{f}_{4\mathtt{u}288}^{\mathtt{u}}; \mathtt{f}_{5\mathtt{u}576} \leftarrow \mathtt{f}_{5\mathtt{u}288}^2;$
//list positive powers of f
$\mathtt{w}_0 \leftarrow \mathtt{f}_{2\mathtt{u}54}\mathtt{f}_{4\mathtt{u}360}; \mathtt{w}_1 \leftarrow \mathtt{f}_{2\mathtt{u}66}\mathtt{f}_{4\mathtt{u}360}; \mathtt{w}_2 \leftarrow \mathtt{f}_7\mathtt{f}_{2\mathtt{u}138}\mathtt{f}_{4\mathtt{u}360}; \mathtt{w}_3 \leftarrow \mathtt{f}_{\mathtt{u}5}\mathtt{f}_{3\mathtt{u}204}\mathtt{f}_{5\mathtt{u}576};$
$\mathtt{y}_0 \leftarrow \mathtt{w}_0\mathtt{w}_1^{\mathtt{p}}\mathtt{w}_2^{\mathtt{p}^2}\mathtt{w}_3^{\mathtt{p}^3};$
//list negative powers of f
$\mathtt{z}_0 \leftarrow \mathtt{f}_{\mathtt{u}6}\mathtt{f}_{3\mathtt{u}162}\mathtt{f}_{5\mathtt{u}288}; \mathtt{z}_1 \leftarrow \mathtt{f}_{\mathtt{u}13}\mathtt{f}_{3\mathtt{u}210}\mathtt{f}_{5\mathtt{u}288}; \mathtt{z}_2 \leftarrow \mathtt{f}_{\mathtt{u}39}\mathtt{f}_{3\mathtt{u}306}\mathtt{f}_{5\mathtt{u}288};$
$\mathtt{z}_3 \leftarrow \mathtt{f}_1\mathtt{f}_{2\mathtt{u}54}\mathtt{f}_{4\mathtt{u}432}; \mathtt{y}_1 \leftarrow \mathtt{z}_0\mathtt{z}_1^{\mathtt{p}}\mathtt{z}_2^{\mathtt{p}^2}\mathtt{z}_3^{\mathtt{p}^3}; \mathtt{g}_1 \leftarrow 1/\mathtt{y}1;$
return $\mathtt{y}_0\mathtt{g}_1;$

Algorithm 4 Final exponentiation for Family 23 with $u > 0$.

Input: Some $\mathtt{u} \in \mathbb{Z}_{>0}$ such that $\mathtt{p} = \mathtt{p}(\mathtt{u})$ and $\mathtt{r} = \mathtt{r}(\mathtt{u})$ are prime; some $\mathtt{f} \in \mathbb{F}_{\mathtt{p}^{16}}$

Output: The value $\mathtt{f}^{\frac{\mathtt{p}^{16}-1}{\mathtt{r}}}$

$\mathtt{t} \leftarrow \mathtt{f}^{\mathtt{p}^6}; \mathtt{f} \leftarrow \mathtt{t}/\mathtt{f};$
$\mathtt{u}_1 \leftarrow \mathtt{u}/2; \mathtt{g}_1 \leftarrow \mathtt{f}^{\mathtt{u}_1}; \mathtt{g}_2 \leftarrow \mathtt{g}_1^{\mathtt{u}_1};$
$\mathtt{f}_{\mathtt{u}1} \leftarrow \mathtt{f}^{\mathtt{u}}; \mathtt{f}_{\mathtt{u}2} \leftarrow \mathtt{f}_{\mathtt{u}1}^{\mathtt{u}}; \mathtt{f}_{\mathtt{u}3} \leftarrow \mathtt{f}_{\mathtt{u}2}^{\mathtt{u}}; \mathtt{f}_{\mathtt{u}4} \leftarrow \mathtt{f}_{\mathtt{u}3}^{\mathtt{u}}; \mathtt{f}_{\mathtt{u}5} \leftarrow \mathtt{f}_{\mathtt{u}4}^{\mathtt{u}}; \mathtt{f}_{\mathtt{u}6} \leftarrow \mathtt{f}_{\mathtt{u}5}^{\mathtt{u}};$
$\mathtt{y} \leftarrow \mathtt{f}_{\mathtt{u}6}\mathtt{g}_2\mathtt{f};$
$\mathtt{t}_0 \leftarrow 1;$
for $\mathtt{i} = 7$ to 1 **do**
    $\mathtt{t} \leftarrow \mathtt{y}^{\mathtt{p}^{\mathtt{i}}}; \mathtt{t}_0 \leftarrow \mathtt{t}_0\mathtt{t}; \mathtt{y} \leftarrow \mathtt{y}^{\mathtt{u}};$
$\mathtt{t}_0 \leftarrow \mathtt{t}_0\mathtt{y}\mathtt{f};$
return $\mathtt{t}_0$;

Algorithm 5 Final exponentiation for Family 25 with $u < 0$.

Input: Some $\mathtt{u} \in \mathbb{Z}_{<0}$ such that $\mathtt{p} = \mathtt{p}(\mathtt{u})$ and $\mathtt{r} = \mathtt{r}(\mathtt{u})$ are prime; some $\mathtt{f} \in \mathbb{F}_{\mathtt{p}^{18}}$

Output: The value $\mathtt{f}^{\frac{\mathtt{p}^{18}-1}{\mathtt{r}}}$

$\mathtt{u} \leftarrow |\mathtt{u}|$
$\mathtt{t} \leftarrow \mathtt{f}^{\mathtt{p}^9}; \mathtt{f} \leftarrow \mathtt{t}/\mathtt{f}; \mathtt{f} \leftarrow \mathtt{f}^{\mathtt{p}^3}\mathtt{f};$
$\mathtt{f}_{1\mathtt{u}} \leftarrow \mathtt{f}^{\mathtt{u}}; \mathtt{f}_{2\mathtt{u}} \leftarrow \mathtt{f}_{1\mathtt{u}}^{\mathtt{u}}; \mathtt{f}_{3\mathtt{u}} \leftarrow \mathtt{f}_{2\mathtt{u}}^{\mathtt{u}}; \mathtt{f}_{4\mathtt{u}} \leftarrow \mathtt{f}_{3\mathtt{u}}^{\mathtt{u}}; \mathtt{f}_{5\mathtt{u}} \leftarrow \mathtt{f}_{4\mathtt{u}}^{\mathtt{u}}; \mathtt{f}_{6\mathtt{u}} \leftarrow \mathtt{f}_{5\mathtt{u}}^{\mathtt{u}};$
$\mathtt{f}_{1\mathtt{u}2} \leftarrow \mathtt{f}_{1\mathtt{u}}^2; \mathtt{f}_{6\mathtt{u}2} \leftarrow \mathtt{f}_{6\mathtt{u}}^2; \mathtt{f}_{6\mathtt{u}3} \leftarrow \mathtt{f}_{6\mathtt{u}2}\mathtt{f}_{6\mathtt{u}};$
$\mathtt{f}_2 \leftarrow \mathtt{f}^2; \mathtt{f}_3 \leftarrow \mathtt{f}_2\mathtt{f}; \mathtt{f}_4 \leftarrow \mathtt{f}_2^2; \mathtt{g}_3 \leftarrow 1/\mathtt{f}_3;$
$\mathtt{y} \leftarrow \mathtt{f}_4\mathtt{f}_{1\mathtt{u}2}\mathtt{f}_{2\mathtt{u}}\mathtt{f}_{6\mathtt{u}3}; \mathtt{y}_{1\mathtt{u}} \leftarrow \mathtt{y}^{\mathtt{u}}; \mathtt{y}_{2\mathtt{u}} \leftarrow \mathtt{y}_{1\mathtt{u}}^{\mathtt{u}};$
$\mathtt{z} \leftarrow \mathtt{g}_3\mathtt{y}_{2\mathtt{u}}; \mathtt{z}_{1\mathtt{u}} \leftarrow \mathtt{z}^{\mathtt{u}}; \mathtt{z}_{2\mathtt{u}} \leftarrow \mathtt{z}_{1\mathtt{u}}^{\mathtt{u}}; \mathtt{z}_{3\mathtt{u}} \leftarrow \mathtt{z}_{2\mathtt{u}}^{\mathtt{u}};$
$\mathtt{w} \leftarrow 1/\mathtt{y}; \mathtt{x} \leftarrow 1/\mathtt{z}_{3\mathtt{u}};$
$\mathtt{w}_0 \leftarrow \mathtt{y}; \mathtt{w}_1 \leftarrow (\mathtt{g}_3\mathtt{x})^{\mathtt{p}}; \mathtt{w}_2 \leftarrow \mathtt{y}_{1\mathtt{u}}^{\mathtt{p}^2}; \mathtt{w}_3 \leftarrow \mathtt{z}_{1\mathtt{u}}^{\mathtt{p}^3}; \mathtt{w}_4 \leftarrow (\mathtt{y}_{2\mathtt{u}}\mathtt{z}_{3\mathtt{u}})^{\mathtt{p}^4}; \mathtt{w}_5 \leftarrow \mathtt{z}_{2\mathtt{u}}^{\mathtt{p}^5}$
$\mathtt{t}_0 \leftarrow \mathtt{w}_0\mathtt{w}_1\mathtt{w}_2\mathtt{w}_3\mathtt{w}_4\mathtt{w}_5;$
return $\mathtt{t}_0$;

B Computing the cost of the Miller loop

There is lot of literature on the optimization of the Miller loop [3,11,13]. Our three recommendations are for curves with quartic and sextic twists, for which the curve forms Jacobi Quartic and Weierstrass are the most efficient; here follows a table with the most efficient formulae for Miller's algorithm in these cases to date. As, in the case of Jacobi Quartic curves, the curve constants are typically very large, we have assumed here that multiplication by a constant in $\mathbb{F}_{p^i}$ with an element in $\mathbb{F}_{p^i}$ costs $\mathbf{m}_i$.

Table 9. Cost of Miller's algorithm for different curve shapes

Twist degree	6	4	4
Curve form	Weierstrass	Weierstrass	Jacobi Quartic
Cost $h_{R,R}(P)$	$2\mathbf{m}_{\frac{k}{6}} + 7\mathbf{s}_{\frac{k}{6}} + \frac{k}{3}\mathbf{m}_1$	$2\mathbf{m}_{\frac{k}{4}} + 8\mathbf{s}_{\frac{k}{4}} + (\frac{k}{2})\mathbf{m}_1$	$4\mathbf{m}_{\frac{k}{4}} + 7\mathbf{s}_{\frac{k}{4}} + \frac{k}{2}\mathbf{m}_1$
Cost $f^2 \cdot h$	$\mathbf{s}_k + 13\mathbf{m}_{\frac{k}{6}}$	$s_k + 8\mathbf{m}_{\frac{k}{4}}$	$\mathbf{s}_k + 8\mathbf{m}_{\frac{k}{4}}$
Cost $h_{R,P}(P)$	$10\mathbf{m}_{\frac{k}{6}} + 2\mathbf{s}_{\frac{k}{6}} + \frac{k}{3}\mathbf{m}_1$	$2\mathbf{m}_{\frac{k}{4}} + 8\mathbf{s}_{\frac{k}{4}} + \frac{k}{2}\mathbf{m}_1$	$13\mathbf{m}_{\frac{k}{4}} + 7\mathbf{s}_{\frac{k}{4}} + \frac{k}{2}\mathbf{m}_1$
Cost $f \cdot h$	$13\mathbf{m}_{\frac{k}{6}}$	$8\mathbf{m}_{\frac{k}{4}}$	$8\mathbf{m}_{\frac{k}{4}}$
Reference	[11]	[11]	[13]

C Defining polynomials of Candidate Families

We include here some more details on the families of Tables 1 and 2.

128-bit security level:

1. $k = 8$, $D = 1$, $\rho = 2$, quartic twists, $V = [x, -1, 0, 0]$, $x = 0 \pmod 2$
$p(x) = (x^8 + x^6 + 5x^4 + x^2 + 4x + 4)/4, \quad r(x) = x^4 + 1, \quad t(x) = x^4 + x + 2$

2. $k = 8$, $D = 1$, $\rho = 2$, quartic twists, $V = [x, -1, 0, 0]$, $x = 1 \pmod 2$
$p(x) = (2x^8 - 2x^7 + 3x^6 + 7x^4 - 2x^3 + 3x^2 + 4x + 5)/4, \quad r(x) = x^4 + 1, \quad t(x) = x^4 + x + 2$

3. $k = 8$, $D = 2$, $\rho = 2$, quadratic twists, $V = [x, 0, 0, -1]$, $x = 1 \pmod 2$
$p(x) = (2x^8 + 4x^7 + 3x^6 + 2x^5 + 11x^4 + 12x^3 + 3x^2 + 2x + 9)/8$
$r(x) = x^4 + 1, \quad t(x) = x^4 + x^3 + 2$

4. $k = 8$, $D = 3$, $\rho = 2$, quadratic twists, $V = [x^2, x, 1, 0]$, $x = 1 \pmod 3$
$p(x) = (3x^{16} - 9x^{12} + x^{10} - 2x^9 + 16x^8 - x^6 + 5x^5 - 13x^4 + x^2 - 5x + 7)/3$
$r(x) = x^8 - x^4 + 1, \quad t(x) = x^8 + x^5 - x^4 - x + 2$

5. $k = 8$, $D = 1$, $\rho = 2$, quartic twists, $V = [1, x, 1, 1]$, $x = 3 \pmod 6$
$p(x) = (72x^8 + 12x^7 - 251x^6 - 78x^5 + 1631x^4 + 240x^3 - 2429x^2 - 606x + 6849)/288$
$r(x) = x^4 - 2x^2 + 9, \quad t(x) = (12x^4 + x^3 - 21x^2 - 5x + 117)/12$

6. $k = 8$, $D = 1$, $\rho = 2$, quartic twists, $V = [x, 2, 2, -2]$, $x = 6 \pmod{12}$
$p(x) = (4608x^8 + 96x^7 - 73151x^6 - 2676x^5 + 1624364x^4 + 29280x^3 - 10588304x^2 - 278592x + 96549696)/18432$
$r(x) = x^4 - 8x^2 + 144, \quad t(x) = (96x^4 + x^3 - 762x^2 - 20x + 13896)/96$

7. $k = 8$, $D = 1$, $\rho = 2$, quartic twists, $V = [x + 1, -1, -1, 0]$,
$x = 2, 14, 20, 26, 44, 50, 56, 62, 68, 74, 80, 86, 92, 98 \pmod{102}$
$p(x) = (9x^8 + 60x^7 + 223x^6 + 590x^5 + 1175x^4 + 1756x^3 + 1963x^2 + 1578x + 666)/36$
$r(x) = x^4 + 4x^3 + 8x^2 + 12x + 9, \quad t(x) = (12x^4 + 47x^3 + 48x^2 + 8x + 112)/12$

8. $k = 8$, $D = 1$, $\rho = 2$, quartic twists, $V = [x + 1, -1, -1, 0]$, $x = 3 \pmod 6$

$p(x) = (72x^8+564x^7+1681x^6+2352x^5+2876x^4+5656x^3+5416x^2+928x+6304)/288$
$r(x) = x^4+4x^3+4x^2+8, \quad t(x) = (12x^4+47x^3+42x^2-4x+112)/12$

9. $k=8$, $D=1$, $\rho=2$, quartic twists, V: could not find a useful short vector
$x = 10, 60 \pmod{70}$
$p(x) = (1225x^{16}-56x^{14}+117601x^{12}-4320x^{10}+4356173x^8-113448x^6+73619425x^4-1023276x^2+480062500)/4900$
$r(x) = x^8+48x^4+625, \quad t(x) = (175x^8-4x^6+8400x^4-117x^2+109550)/175$

10. $k=9$, $D=3$, $\rho=2$, cubic twists, $V=[x,0,-1,0,0,0]$, $x = 2 \pmod 3$
$p(x) = (3x^{12}+3x^{11}+x^{10}+9x^9+7x^8+x^7+16x^6+10x^5+x^4+13x^3+4x^2+7)/3$
$r(x) = x^6+x^3+1, \quad t(x) = x^6+x^5+x^3+2$

11. $k=9$, $D=3$, $\rho=2$, cubic twists, $V=[x-1,-2,1,0,0,0]$, $x = 2 \pmod 3$
$p(x) = 29241x^{12}-350721x^{11}+1576798x^{10}-2506904x^9-1833891x^8-88143x^7+59402835x^6-144152175x^5+18500997x^4+173184927x^3+464941830x^2-1565297066x+1150985383)/29241$
$r(x) = x^6-6x^5+9x^4+11x^3-6x^2-135x+199, \quad t(x) = (171x^6-1021x^5+1522x^4+1883x^3-964x^2-22916x+33737)/171$

12. $k=9$, $D=3$, $\rho=2$, cubic twists, $V=[x,0,3,0,0,2]$, $x = 7 \pmod{21}$
$p(x) = (147x^{12}-24x^{11}+x^{10}+10731x^9-1322x^8+37x^7+294049x^6-24299x^5+343x^4+3584644x^3-149048x^2+16403632)/147$
$r(x) = x^6+37x^3+343, \quad t(x) = (49x^6-3x^5+1813x^3-62x^2+16856)/49$

13. $k=10$, $D=1$, $\rho=1.75$, quadratic twists, $V=[x^2,-1,0,0]$, $x = 1 \pmod 2$
$p(x) = (x^{14}-2x^{12}+x^{10}+x^4+2x^2+1)/4, \quad r(x) = x^8-x^6+x^4-x^2+1, \quad t(x) = x^2+1$

14. $k=10$, $D=5$, $\rho=1.75$, quadratic twists, $V=[x^2-1,1,-1,1]$, $x = \{0,4,6\} \pmod{10}$
$p(x) = (4x^{14}-7x^{12}+11x^{10}-11x^8-9x^6+13x^4-16x^2+20)/20$
$r(x) = x^8-x^6+x^4-x^2+1, \quad t(x) = x^2+1-x^6+x^4-x^2+2$

15. $k=10$, $D=15$, $\rho=1.75$, quadratic twists, $V=[x,0,-1,x^2]$, $x = \{1,3,6,13\} \pmod{15}$
$p(x) = (4x^{14}+4x^{13}+x^{12}-12x^{11}-12x^{10}-7x^9+11x^8+17x^7+15x^6-3x^5-11x^4+x^3-2x^2+3x+6)/15$
$r(x) = x^8+x^7-x^5-x^4-x^3+x+1, \quad t(x) = x^3+1$

16. $k=10$, $D=2$, $\rho=1.875$, quadratic twists, $V=[x^4-1,1,-1,1]$, $x = 0 \pmod 4$
$p(x) = (x^{30}-2x^{26}+2x^{24}+x^{22}-2x^{20}+2x^{16}-10x^{12}+x^{10}+10x^8-2x^6-8x^4+x^2+8)/8$
$r(x) = x^{16}-x^{12}+x^8-x^4+1, \quad t(x) = -x^{12}+x^8-x^4+2$

17. $k=12$, $D=3$, $\rho=1.5$, sextic twists, $V=[6x+2,-1,-1,-1]$, $x \in \mathbb{Z}$
$p(x) = 1728x^6+2160x^5+1548x^4+756x^3+240x^2+54x+7, \quad r(x) = 36x^4+36x^3+$

$18x^2 + 6x + 1, \quad t(x) = -6x^2 + 1$

18. $k = 12$, $D = 2$, $\rho = 1.75$, quadratic twists, $V = [x^2, -1, 0, 0]$, $x = 1 \pmod 2$
$p(x) = (x^{14} - 4x^{10} + 2x^8 + 4x^6 - 2x^4 + 5x^2 + 2)/8, \quad r(x) = x^8 - x^4 + 1, \quad t(x) = x^2 + 1$

19. $k = 12$, $D = 3$, $\rho = 1.5$, sextic twists, $V = [x, -2, -1, 1]$, $x = \{8, 23\} \pmod{30}$
$p(x) = (x^6 - 8x^5 + 21x^4 - 17x^3 + 13x^2 + 45x + 21)/225$
$r(x) = x^4 - 2x^3 - 3x^2 + 4x + 13, \quad t(x) = (-x^3 + 4x^2 + 5x + 6)/15$

20. $k = 12$, $D = 3$, $\rho = 1.5$, sextic twists, $V = [2, x, 3, 0]$, $x = \{209, 266\} \pmod{285}$
$p(x) = (x^6 + 8x^5 - 18x^4 - 326x^3 - 342x^2 + 3143x + 6859)/1425$
$r(x) = x^4 - 37x^2 + 361, \quad t(x) = (-2x^3 + 17x + 95)/95$

192-bit security level:
21. $k = 15$, $D = 3$, $\rho = 2$, cubic twists, $V = [x, 0, 0, 0, -1, 0, 0, 0]$, $x = 0 \pmod 3$
$p(x) = (3x^{16} - 9x^{15} + 10x^{14} + 4x^{13} - 19x^{12} + 24x^{11} - 14x^{10} - 6x^9 + 37x^8 - 36x^7 + 8x^6 + 19x^5 - 20x^4 + 21x^3 - 3x^2 - 12x + 12)/3$
$r(x) = x^8 - x^7 + x^5 - x^4 + x^3 - x + 1, \quad t(x) = x^8 - x^7 + x^5 + x^3 - x + 2$

22. $k = 15$, $D = 3$, $\rho = 2$, cubic twists, $V = [3 * x, 0, 0, 0, -1, 0, 0, 0]$, $x \in \mathbb{Z}$
$p(x) = 43046721x^{16} - 14348907x^{15} - 3188646x^{14} + 2125764x^{13} - 708588x^{12} + 177147x^{11} + 78732x^{10} - 59049x^9 + 15309x^8 - 972x^6 + 567x^5 - 54x^4 + 9x^2 - 3x + 1$
$r(x) = 6561x^8 - 2187x^7 + 243x^5 - 81x^4 + 27x^3 - 3x + 1, \quad t(x) = 6561x^8 - 2187x^7 + 243x^5 + 27x^3 - 3x + 2$

23. $k = 16$, $D = 1$, $\rho = 2$, quartic twists, $V = [x, -1, 0, 0, 0, 0, 0, 0]$, $x \equiv 0 \pmod 2$
$p(x) = (x^{16} + x^{10} + 5 * x^8 + x^2 + 4 * x + 4)/4, \quad r(x) = x^8 + 1, \quad t(x) = x^8 + x + 2$

4. $k = 16$, $D = 1$, $\rho = 2$, quartic twists, $V = [2, 0, 0, x, 1, 0, 0, 0]$, $x \equiv \{10, 60\} \pmod{70}$
$p(x) = (245x^{16} + 28x^{13} + 23520x^{12} + x^{10} + 1920x^9 + 871225x^8 + 48x^6 + 45204x^5 + 14723760x^4 + 625x^2 + 361148x + 96012500)/980$
$r(x) = x^8 + 48x^4 + 625, \quad t(x) = (35x^8 + 2x^5 + 1680x^4 + 41x + 21910)/35$

25. $k = 18$, $D = 3$, $\rho = 2$, sextic twists, $V = [x, 1, 0, 0, -1, 0]$, $x \equiv 1 \pmod 3$
$p(x) = (3x^{12} - 3x^9 + x^8 - 2x^7 + 7x^6 - x^5 - x^4 - 4x^3 + x^2 - 2x + 4)/3$
$r(x) = x^6 - x^3 + 1, \quad t(x) = x^6 - x^4 - x^3 + 2$

26. $k = 18$, $D = 3$, $\rho = 2$, sextic twists, $V = [1, 0, x, 2, 0, 0]$, $x \equiv 1 \pmod 3$
$p(x) = (21x^{12} - 6x^{10} + 1533x^9 + x^8 - 334x^7 + 42007x^6 + 37x^5 - 6199x^4 + 512092x^3 + 343x^2 - 38368x + 2343376)/21$
$r(x) = x^6 + 37x^3 + 343, \quad t(x) = (7x^6 + x^4 + 259x^3 + 16x + 2408)/7$

27. $k = 20$, $D = 1$, $\rho = 1.5$, quartic twists, $V = [x, -1, 0, 0, 0, 0, 0, 0]$, $x \equiv 1 \pmod 2$
$p(x) = (x^{12} - 2x^{11} + x^{10} + x^2 + 2x + 1)/4, \quad r(x) = x^8 - x^6 + x^4 - x^2 + 1, \quad t(x) = x + 1$

References

1. Diego F. Aranha, Laura Fuentes-Castañeda, Edward Knapp, Alfred Menezes, and Francisco Rodríguez-Henríquez. Implementing pairings at the 192-bit security level. In *Pairing-Based Cryptography - Pairing 2012 - 5th International Conference, Cologne, Germany, May 16-18, 2012, Revised Selected Papers*, pages 177–195, 2012.
2. Diego F. Aranha, Koray Karabina, Patrick Longa, Catherine H. Gebotys, and Julio López. Faster explicit formulas for computing pairings over ordinary curves. In *Advances in Cryptology – EUROCRYPT 2011*, pages 48–68, 2011.
3. Christophe Arne, Tanja Lange, Michael Naehrig, and Christophe Ritzenthaler. Faster computation of the tate pairing. *Journal of Number Theory*, 131(5):842 – 857, 2011.
4. Razvan Barbulescu and Sylvain Duquesne. Updating key size estimations for pairings. *IACR Cryptology ePrint Archive*, 2017:334, 2017. http://eprint.iacr.org/2017/334.
5. Razvan Barbulescu, Nadia El Mrabet, and Loubna Ghammam. A taxonomy of pairings, their security, their complexity. Cryptology ePrint Archive, Report 2019/485, 2019. https://eprint.iacr.org/2019/485.
6. Paulo S. L. M. Barreto, Craig Costello, Rafael Misoczki, Michael Naehrig, Geovandro C. C. F. Pereira, and Gustavo Zanon. Subgroup security in pairing-based cryptography. In *Progress in Cryptology - LATINCRYPT 2015 - 4th International Conference on Cryptology and Information Security in Latin America, Guadalajara, Mexico, August 23-26, 2015, Proceedings*, pages 245–265, 2015. https://doi.org/10.1007/978-3-319-22174-8_14.
7. Daniel J. Bernstein and Tanja Lange. Explicit-formulas database. http://www.hyperelliptic.org/EFD/.
8. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003. https://doi.org/10.1137/S0097539701398521.
9. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. *J. Cryptology*, 17(4):297–319, 2004. https://doi.org/10.1007/s00145-004-0314-9.
10. Friederike Brezing and Annegret Weng. Elliptic curves suitable for pairing based cryptography. *Des. Codes Cryptography*, 37(1):133–141, 2005. https://doi.org/10.1007/s10623-004-3808-4.
11. Craig Costello, Tanja Lange, and Michael Naehrig. Faster pairing computations on curves with high-degree twists. In *Public Key Cryptography – PKC 2010*, pages 224–242. Springer Berlin Heidelberg, 2010.
12. Augusto Jun Devegili, Colm Ó Héigeartaigh, Michael Scott, and Ricardo Dahab. Multiplication and squaring on pairing-friendly fields. Cryptology ePrint Archive, Report 2006/471, 2006.
13. Sylvain Duquesne, Nadia El Mrabet, and Emmanuel Fouotsa. Efficient computation of pairings on jacobi quartic elliptic curves. *Journal of Mathematical Cryptology*, 8(4):331–362, 2014. https://hal.archives-ouvertes.fr/hal-01095359/document.
14. Georgios Fotiadis and Elisavet Konstantinou. TNFS resistant families of pairing-friendly elliptic curves. *IACR Cryptology ePrint Archive*, 2018:1017, 2018. https://eprint.iacr.org/2018/1017.

15. Georgios Fotiadis and Chloe Martindale. Optimal tnfs-secure pairings on elliptic curves with even embedding degree. Cryptology ePrint Archive, Report 2018/969, 2018. https://eprint.iacr.org/2018/969.
16. Gerhard Frey and Hans-Georg Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Mathematics of computation*, 62(206):865–874, 1994.
17. Laura Fuentes-Castañeda, Edward Knapp, and Francisco Rodríguez-Henríquez. Faster hashing to $\mathbb{G}_2$. In *Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers*, pages 412–430, 2011. https://doi.org/10.1007/978-3-642-28496-0_25.
18. Robert Granger and Michael Scott. Faster squaring in the cyclotomic subgroup of sixth degree extensions. In *Public Key Cryptography – PKC 2010*, pages 209–223, 2010.
19. Trusted Computing Group. TPM 2.0 library specification, 2016. https://trustedcomputinggroup.org/resource/tpm-library-specification/.
20. Aurore Guillevic, Simon Masson, and Emmanuel Thome. Cocks-pinch curves of embedding degrees five to eight and optimal ate pairing computation. Cryptology ePrint Archive, Report 2019/431, 2019. https://eprint.iacr.org/2019/431.
21. Antoine Joux. A one round protocol for tripartite diffie–hellman. *Journal of cryptology*, 17(4):263–276, 2004.
22. Koray Karabina. Squaring in cyclotomic subgroups. *Math. Comput.*, 82(281):555–579, 2013. https://doi.org/10.1090/S0025-5718-2012-02625-1.
23. Taechan Kim and Razvan Barbulescu. Extended tower number field sieve: A new complexity for the medium prime case. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, pages 543–571, 2016.
24. Alfred Menezes, Tatsuaki Okamoto, and Scott A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Trans. Information Theory*, 39(5):1639–1646, 1993. https://doi.org/10.1109/18.259647.
25. Victor S. Miller. The Weil pairing, and its efficient calculation. *J. Cryptology*, 17(4):235–261, 2004. https://doi.org/10.1007/s00145-004-0315-8.
26. Frederik Vercauteren. Optimal pairings. *IEEE Trans. Information Theory*, 56(1):455–461, 2010. https://doi.org/10.1109/TIT.2009.2034881.