A taxonomy of pairings, their security, their complexity

Razvan Barbulescu[1], Nadia El Mrabet[2], and Loubna Ghammam[3]

[1] CNRS, University of Bordeaux, France
razvan.barbulescu@u-bordeaux.fr
[2] Mines Saint-Etienne, CEA-Tech, Centre CMP, Departement SAS, France
nadia.el-mrabet@emse.fr
[3] ITK-Engineering, Germany
loubna.ghammam@itk-engineering.de

Abstract. The Kim-Barbulescu attack against pairings made it necessary to increase the key sizes of the most popular families of pairings : BN, BLS-12, KSS-16, KSS-18 and BLS-24. The computation of new key sizes was a slow process because it was done in two waves : first a series of theoretical estimations, then a wave of precise estimations based on practical models BD model [5] and GS [40]. In this paper, we propose an up-to-date security evaluation for more then hundred pairing friendly elliptic curves. We evaluate the complexity of a complete pairing execution taking into account the Miller algorithm for different degree of twist and the Final exponentiation for the most promising curves. At 128 bits of security we find that the best pairings in the BD model are BLS-24 and BLS-12. The best pairings are not affected by the new polynomial selection method [40]. At 192 bits of security, we find that the new champions are the less known BLS-24, KSS-16 and KSS-18. At 256 bits of security we conclude that the best pairing is BLS-27.

1 Introduction

Pairings are a crucial ingredient in a series of public-key protocols which started with Joux' [44] tripartite Diffie-Hellman scheme and Bonkeh and Franklin's Franklin [14] identity-based encryption. Then followed protocols for short signatures [17], a wide variety of aggregate, instance and verifier-local revocation signatures [15,13,47], broadcast encryption [16], cloud computing [4], privacy enhancing environments [76], deep package inspection over encrypted traffic [77,18] and many others. The NIST [64] pilots a project dedicated to pairings. Efficient implementations of pairings [12], [11], [38], [79], [49] made them interesting for industrial development [78,19,1].
Pairings are not suited for post-quantum applications as they are based on the difficulty of discrete logarithms. However, quantum computer is not readily available for large computations and one can continue to use pairings for applications where the keys are used for a short period of time.
This paper is the mature result of a work we started a few years ago in which we reacted to the Kim-Barbulescu TNFS attack. Before the attack, the security of pairings was a function of the key sizes, regardless on which family of pairings was used. In that context, the fastest pairings were BN, BLS12, KSS16 at the 128 bits security level, KSS18 and BLS24 for higher security which had small values of a parameter called ρ. A recent article [5] showed that these pairings are affected by the TNFS attack. We raised the question whether there are families which are less affected and which become the new champions and whether the existing order is reshuffled. For this we worked over hundred families of pairings. A precise analysis allows us to make the following recommendations, which confirm that the order has changed and that there are new champions:

- 128 bits : BLS-24 and BLS-12 are the champions, followed by k10m62, KSS 16 and DCC 15 families;
- 192 bits : BLS-24 is the champion, followed by KSS-16 and KSS-18;

– 256 bits : BLS-27 is the clear champion.

At a high level, a pairing is a non-degenerate and bilinear map, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are subgroups of an elliptic curve and $\mathbb{G}_3$ is a multiplication sub-group of a finite field. The security of pairing-based cryptography relies on one side on the discrete logarithm problem (DLP) over $\mathbb{G}_1$ (and consecutively over $\mathbb{G}_2$) which are elliptic curves, we call this the curve side security and note that it is very well understood on the classical computers. On the other side, it relies on the discrete logarithm problem over $\mathbb{G}_3$ which is the multiplicative group of a finite field, this is the field side security.

The hardness of computing discrete logarithms in a finite field is difficult to evaluate. In a first time one used the approximation that its cost is equal to that of factoring, which is done with a variant of the same algorithm : the number field sieve (NFS). Hence, the first key sizes proposed for pairings [56] were such that $\log_2 \#\mathbb{G}_3$ matches the required bit size for an RSA module offering the same security level. In a second time, one computed the cost using a theoretical upper bound [61],[74] and the recommended key sizes were used go generate new seeds [29],[80] and to propose efficient implementations [33]. In a recent article, Barbulescu and Duquesne [5] made a precise real-life analysis with no theoretical assumption. Hence, they found the optimal parameters for each variant of NFS and obtained key sizes which can be used in a future standardization for 5 families of pairing friendly elliptic curves. All the recent works use the practical estimation : we used it in a working version of this article [8], Martindale and Fotiadis [30,31] used it to compute the security of pairings whose embedding degree is even and respectively composite, Guillevic [40] made a short list at 128 its of security and Guillevic and Singh [42] used it in a preliminary presentation for some families at 192 bits of security.

The core of the security analysis is actually the daily difficulty for a NFS implementer and developer : select polynomials and tune parameters. This task is not automatic in the CADO-NFS software as the tabulated parameters for factoring integers between 80 and 100 decimal digits are not guaranteed to be optimal. It is an open question to rapidly select the optimal parameters for NFS, especially for the smallest values where NFS is the choice algorithm, i.e. integers of about 80 decimal digits. In our case however, of 128 bits of security, tuning parameters is negligible and we used a brute force approach : we test a wide range of NFS variants and parameters and experimentally measure and extrapolate the cost of an NFS computation until we find the optimal set of parameters. Guillevic found errors in the first version of this article because the range of parameters was not wide enough. We solved this by using more computational time and by launching computations in an automatic manner.

Once the security has been settled, we continue by finding seeds of small NAF weight and optimize as much as possible the computation of the Ate pairing : Miller's loop and final exponentiation. To shorten presentation we use BN as a broom wagon at 128 bits of security and similarly for the higher levels : if Miller's loop of a pairing costs more than the complete computation for BN then we discard this pairing from the final exponentiation optimization.

Our contribution

We make an extensive literature inspection to find as many pairing-friendly families as possible. The main reference is the taxonomy [34] whose title we copy, but we discovered some families [24],[57] which weren't included in that work. We also add a small number of families which were published after the taxonomy : [23],[74]. Before the key sizes had to be corrected, the BN family was much faster and received much more attention than the other families in the taxonomy, some of which remained to the status of theoretical formulae.

We continue along the lines of the recent works and make the precise estimations of the security for a large number of families. In the case of the families studied in [40] and [42] the authors used a slightly different model which results in key sizes which are within an error of 5% from the model of [5] and [31]. For a fair comparison we compute here the key sizes in the same model as BN, BLS and KSS. We present the first precise analysis at 256 bits of security.

We evaluate the complexity of the pairings at the sizes which resist to the new attacks. In the case of pairings whose embedding degree is divisible by 5, 7 or 11 we discuss the Karatsuba-like formulae introduced in [27] and [71].

Paper overview

In Section 2, we recall the basic notations on pairings, present the classical optimizations of the implementation and recall the various constructions of pairings. In Section 3, we draw the big lines of the NFS algorithm, recall what are the choices for an attacker and compute the updated key sizes for a large number of families. For each family, we construct pairings and evaluate the cost of Miller's loop, first in arithmetic then in binary operations, at 128 bits (Section 4) and respectively 192 and 256 bits of security (Section 4.9 and 4.10). Then, in Section 5 we present the final exponentiation complexity for the Optimal Ate pairings in some of proposed curves. We obtain the overall cost and conclude in Section 6, the result tables are at the end of the article.

2 Some background on pairings

Pairings are a crucial ingredient in a series of public-key protocols which started with Joux' [44] tripartite Diffie-Hellman scheme and Bonkeh and Franklin's Franklin [14] identity-based encryption. Then followed protocols for short signatures [17], a wide variety of aggregate, instance and verifier-local revocation signatures [15,13,47], broadcast encryption [16], cloud computing [4], privacy enhancing environments [76], deep package inspection over encrypted traffic [77,18] and many others. The NIST [64] pilots a project dedicated to pairings. Efficient implementations of pairings [12], [11], [38], [79], [49] made them interesting for industrial development [78,19].
Pairings are not suited for post-quantum applications as they are based on the difficulty of discrete logarithms. However, quantum computer is not readily available for large computations and one can continue to use pairings for applications where the keys are used for a short period of time.
This paper is the mature result of a work we started a few years ago in which we reacted to the Kim-Barbulescu TNFS attack. Before the attack, the security of pairings was a function of the key sizes, regardless on which family of pairings was used. In that context, the fastest pairings were BN, BLS12, KSS16 at the 128 bits security level, KSS18 and BLS24 for higher security which had small values of a parameter called ρ. A recent article [5] showed that these pairings are affected by the TNFS attack. We raised the question whether there are families which are less affected and which become the new champions and whether the existing order is reshuffled. For this we worked over hundred families of pairings. A precise analysis allows us to make the following recommendations, which confirm that the order has changed and that there are new champions:

- 128 bits : BLS-24 and BLS-12 are the champions, followed by k10m62, KSS 16 and DCC 15 families;
- 192 bits : BLS-24 is the champion, followed by KSS-16 and KSS-18;
- 256 bits : BLS-27 is the clear champion.

At a high level, a pairing is a non-degenerate and bilinear map, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are subgroups of an elliptic curve and $\mathbb{G}_3$ is a multiplication sub-group of a finite field. The security of pairing-based cryptography relies on one side on the discrete logarithm problem (DLP) over $\mathbb{G}_1$ (and consecutively over $\mathbb{G}_2$) which are elliptic curves, we call this the curve side security and note that it is very well understood on the classical computers. On the other side, it relies on the discrete logarithm problem over $\mathbb{G}_3$ which is the multiplicative group of a finite field, this is the field side security.
The hardness of computing discrete logarithms in a finite field is difficult to evaluate. In a first time one used the approximation that its cost is equal to that of factoring, which is done with a variant of the same algorithm : the number field sieve (NFS). Hence, the first key sizes proposed for pairings [56] were such that $\log_2 \#\mathbb{G}_3$ matches the required bit size for an RSA module offering the same security level. In a second time, one computed the cost using a theoretical upper bound [61],[74] and the recommended key sizes were used go generate new seeds [29],[80] and to propose efficient implementations [33]. In a recent article, Barbulescu and

Duquesne [5] made a precise real-life analysis with no theoretical assumption. Hence, they found the optimal parameters for each variant of NFS and obtained key sizes which can be used in a future standardization for 5 families of pairing friendly elliptic curves. All the recent works use the practical estimation : we used it in a working version of this article [8], Martindale and Fotiadis [30,31] used it to compute the security of pairings whose embedding degree is even and respectively composite, Guillevic [40] made a short list at 128 its of security and Guillevic and Singh [42] used it in a preliminary presentation for some families at 192 bits of security.
The core of the security analysis is actually the daily difficulty for a NFS implementer and developer : select polynomials and tune parameters. This task is not automatic in the CADO-NFS software as the tabulated parameters for factoring integers between 80 and 100 decimal digits are not guaranteed to be optimal. It is an open question to rapidly select the optimal parameters for NFS, especially for the smallest values where NFS is the choice algorithm, i.e. integers of about 80 decimal digits. In our case however, of 128 bits of security, tuning parameters is negligible and we used a brute force approach : we test a wide range of NFS variants and parameters and experimentally measure and extrapolate the cost of an NFS computation until we find the optimal set of parameters. Guillevic found errors in the first version of this article because the range of parameters was not wide enough. We solved this by using more computational time and by launching computations in an automatic manner.
Once the security has been settled, we continue by finding seeds of small NAF weight and optimize as much as possible the computation of the Ate pairing : Miller's loop and final exponentiation. To shorten presentation we use BN as a broom wagon at 128 bits of security and similarly for the higher levels : if Miller's loop of a pairing costs more than the complete computation for BN then we discard this pairing from the final exponentiation optimization.

Our contribution

We make an extensive literature inspection to find as many pairing-friendly families as possible. The main reference is the taxonomy [34] whose title we copy, but we discovered some families [24],[57] which weren't included in that work. We also add a small number of families which were published after the taxonomy : [23],[74]. Before the key sizes had to be corrected, the BN family was much faster and received much more attention than the other families in the taxonomy, some of which remained to the status of theoretical formulae.
We continue along the lines of the recent works and make the precise estimations of the security for a large number of families. In the case of the families studied in [40] and [42] the authors used a slightly different model which results in key sizes which are within an error of 5% from the model of [5] and [31]. For a fair comparison we compute here the key sizes in the same model as BN, BLS and KSS. We present the first precise analysis at 256 bits of security.
We evaluate the complexity of the pairings at the sizes which resist to the new attacks. In the case of pairings whose embedding degree is divisible by 5, 7 or 11 we discuss the Karatsuba-like formulae introduced in [27] and [71].

Paper overview

In Section 2, we recall the basic notations on pairings, present the classical optimizations of the implementation and recall the various constructions of pairings. In Section 3, we draw the big lines of the NFS algorithm, recall what are the choices for an attacker and compute the updated key sizes for a large number of families. For each family, we construct pairings and evaluate the cost of Miller's loop, first in arithmetic then in binary operations, at 128 bits (Section 4) and respectively 192 and 256 bits of security (Section 4.9 and 4.10). Then, in Section 5 we present the final exponentiation complexity for the Optimal Ate pairings in some of proposed curves. We obtain the overall cost and conclude in Section 6, the result tables are at the end of the article.

2.1 Arithmetic for finite fields

Notations In the following we use the classical notations A_q, M_q, S_k and I_q for the binary cost of the addition, multiplication, squaring and respectively inversion over $\mathbb{F}_q$. We denote by M_k, S_k and I_k the binary cost of the multiplication, squaring and inversion in the field $\mathbb{F}_{q^k}$. For our level of optimization, the crude estimation $M = S$ is enough. When a multiplication by an element of $\mathbb{F}_q$ is necessary (for instance a multiplication by a, denoted d_a, in the doubling of points) we make the coarse estimation that $d_a = M_q$. We call **D**, **A**, M**A** and **L** the cost of a doubling, addition and mixed addition on the elliptic curve and respectively a final line evaluation.

Arithmetic The complexity of the multiplication M_k is a very challenging task in pairing-based cryptography. Several papers present optimized algorithms for the extension over a finite field [54,63,27,2,81,25,75,67]. Of course, the schoolbook method can always be applied but for a value k, the complexity of M_k is $k^2 M_q$. The tricks for multiplication in F_{p^k} are made to decrease the number of multiplications in M_q, but as *all magic comes with a price* the tricks increase the number of additions in M_q. The ratio $R = \frac{M_q}{A_q}$ is then the balance precising if we could use a method over another.
In Table 1 we recall classical complexities that are used in pairings, then we summarize results from the literature [63,28,27] which are asymptotically better but are yet to prove there efficiency in the pairings implementations. These latter formulae often achieve the mathematical lower bound for the number of M_q with the cost of increasing the number of A_q. We use the inequalities $M_{11} \leq M_{12}$, $M_{13} \leq M_{14}$, $M_{17} \leq M_{18}$ and $M_{19} \leq M_{20}$.

Classical exponents									
extension	$\mathbb{F}_{p^2}$	$\mathbb{F}_{p^3}$	$\mathbb{F}_{p^4}$	$\mathbb{F}_{p^{12}}$	$\mathbb{F}_{p^{16}}$	$\mathbb{F}_{p^{18}}$	$\mathbb{F}_{p^n}$		
M_k/M_q	$3M$	$5\ M$	$9M$	$54M$	81	108	$\frac{n(n+1)}{2}$		
non classical exponents									
extention	$\mathbb{F}_{p^5}$	$\mathbb{F}_{p^6}$	$\mathbb{F}_{p^7}$	$\mathbb{F}_{p^{11}}$	$\mathbb{F}_{p^{13}}$	$\mathbb{F}_{p^{14}}$	$\mathbb{F}_{p^{15}}$	$\mathbb{F}_{p^{17}}$	$\mathbb{F}_{p^{19}}$; $\mathbb{F}_{p^{20}}$
M_k/M_q Upper bound	13	17	22	46	49	53	75	94	105
M_k/M_q Lower bound	9	11	13	33	39	39	45	65	99

Table 1: Optimized complexities of the multiplication over extension fields

We go from the arithmetic complexity to the binary complexity using the crude estimate that M_q counts for w^2 word multiplications, where w is the number of machine words of q. We denote by m_{32} (resp. m_{64}) the cost of a word multiplication on a 32-bit (resp 64-bit machine). A comparison of hardware implementation is beyond the scope of this article because it is much more difficult to take into account the dedicated architectures.

2.2 Cost of Miller's loop

The Miller loop is a double-and-add algorithm similar to the fast exponentiation. Hence it consists in a number of iterations of the doubling and addition step, plus a final line evaluation. A doubling step followed by an addition step can be done together in a mixed step. The complexity of each step depends on two parameters : the twist of the elliptic curve and the choice of coordinates, as we summarize in Table 2.

Operation	Complexity	
Twist	Sextic twist	Quadratic twist
Doubling [20]	$(2k/d)M_q + 3M_e + 5S_e + M_k + S_k$	$(2k/d)M_q + 2M_e + 8S_e + 1d_a + M_k + S_k$
Addition [20]	$(2k/d)M_q + 14M_e + 2S_e + 1d_c + M_k$	$(2k/d)M_q + 12M_e + 7S_e + M_k$
Mixed add [20]	$(2k/d)M_q + 10M_e + 2S_e + 1d_c + M_k$	$(2k/d)M_q + 9M_e + 5S_e + M_k$
Final line eval.	$2k/dM_q + 5M_e$	$5M_e + 2k/dM_q$ [5]

Operation	Complexity	
Twist	Cubic twist	Quartic twist
Doubling	$M_{3b} + kM_q + 3M_e + 9S_e + M_k + S_k$ [81]	$(2k/d)M_q + 3M_e + 6S_e + M_k + S_k$ [5]
Mixed	$kM_q + 12M_e + 5S_e + M_k$ [81]	$(2k/d)M_q + 9M_e + 5S_e + M_k$ [5]
Final line eval.	$(5k-4)M_q + S_q + S_{k/d} + M_{k/d} + 2M\mathbf{A}$	$5M_e + 2k/dM_q$ [5]

Table 2: Complexity of Miller's steps using twists

2.3 Expression of Optimal Ate pairing

The expression of the Optimal Ate pairing is obtained after the reduction of a lattice constructed using the polynomial expression of $q(x)$ and $r(x)$. As a consequence, for each method of construction, we have a specific equation for the Optimal Ate pairing. There are constructions where the value of k also changes the expression of the Optimal Ate. We present in Table 3 the simple expression of the Miller loop for the Optimal Ate pairing depending on the method of construction and the embedding degree. By raising the Miller expression to power $\frac{q^k-1}{r}$ one obtains the expression of the Optimal Ate pairing. When the Optimal Ate pairing is not uniquely defined we refer to the subsection where the reader can find the details on the formulae.

3 Overview of the NFS attacks

The extended tower number field sieve, exTNFS, encompasses all the variants of NFS : NFS, SNFS, exTNFS-Conj, SexTNFS-JP etc. Let us present briefly the algorithm with a special care on the choices that can be made by an attacker.

3.1 Big lines of the algorithm

At a high level, exTNFS on $\mathbb{F}_{q^k}$ proceeds as follows. Let κ and η be two divisors of k so that $k = \kappa\eta$. Let $h(t)$ be a polynomial of degree η in $\mathbb{Z}[t]$ which is irreducible modulo q, and call ω a root of $h(t)$ in $\mathbb{F}_q[t]/\langle h\rangle$. Then select two polynomials $f(t,x)$ and $g(t,x)$ in $\mathbb{Z}[t,x]$ such that $f(\omega,x)$ and $g(\omega,x)$ have a common irreducible factor of degree κ in $\mathbb{F}_q(\omega) = \mathbb{F}_{q^\eta}$. This step, called polynomial selection, takes a negligible time but determines the cost of the whole algorithm.
In the sieving stage, for a given parameter A, one considers the pairs $(a(t), b(t)) \in \mathbb{Z}[t]^2$ of degree less than η such that $\max(\|a\|_\infty, \|b\|_\infty) \leq A$. We call norms of (a,b) the integers $N_f(a,b) = \mathrm{Res}_t(\mathrm{Res}_x(a(t) - xb(t), f(t,x)), h(t))$ and $N_g(a,b) = \mathrm{Res}_t(\mathrm{Res}_x(a(t) - xb(t), g(t,x)), h(t))$. Given a parameter B, the sieving stage outputs the list of (almost) all pairs (a,b) such that $N_f(a,b)$ and $N_g(a,b)$ are B-smooth, i.e. all their prime factors are less than B.
In the linear algebra stage, the goal is to solve a linear system having twice as many elements as primes less than B (the number of prime ideals in the number fields of f and g of norm less than B). This is done in two steps : filtering where the size of the matrix is greatly reduced and the proper linear algebra computations where the obtained linear system is solved. Due to heuristic arguments in [5], the filtering stage reduces the size of the matrix by a factor $\log_2 B$ and the cost of the linear algebra is $2^7 B^2/(\log(B)\log_2 B)^2$.

Construction	Embedding degree for k' an odd integer	Twist	Miller expression in the Optimal Ate
Method 6.2	$k = k'$	–	$\left(f_{x^2,Q}(P) \times \frac{l_{qQ,x^2Q}(P)}{v_{(x^2+q)Q}(P)}\right)$
Method 6.3	$k = 2k'$	2	$\left(f_{x^2,Q}(P) \times l_{-qQ,x^2Q}(P)\right)$
Method 6.4	$k = 4k'$	4	$\left(f_{x,Q}(P) \times l_{-qQ,x^2Q}(P)\right)$
Method 6.6	$k \equiv 0 \bmod 6$	6	$(f_{x,Q}(P) \times l_{-qQ,xQ}(P))$
	$k \equiv 3 \bmod 6$	3	Section 4.4
	$k \equiv 2 \bmod 6$	2	$\left(f_{x^2,Q} f_{x,Q}^q \times l_{s_1Q,x^2Q} \times l_{s_2Q,xqQ}\right)$ $s_0 = x^2 + xq + q^2,\ s_1 = -xq + q^2,\ s_2 = q^2$
	$k \equiv 4 \bmod 6$	2	Section 4.4
	$k \equiv \{1, 5\} \bmod 6$	No	$\left(f_{x^2,Q} f_{-x,Q}^q \frac{l_{s_1Q,x^2Q}}{v_{s_0Q}} \frac{l_{q^2Q,-xqQ}}{v_{s_1Q}}\right)$ $s_0 = x^2 - xq + q^2,\ s_1 = -xq + q^2$
Method 6.7	$k = 12$	2	$(f_{x^2,Q} l_{-qQ,x^2Q})(P)$
Method 6.7	$k = 24$	2	$(f_{x,Q} l_{-pQ,xQ})(P)$
Method 6.7	$k = 18, 30$	2	$(f_{x^4,Q} l_{-qQ,x^4Q})(P)$
Method 6.7	$k = 15$	–	$f_{x^4-1,Q}$+ extra
Method 6.7	$k = 9, 21, 27$	–	$f_{x^4,Q}(P) \frac{l_{q^5Q,x^4Q}(P)}{v_{[x^4+q^5]Q}(P)}$
KSS	even	4, 6	Section 4.7
Other families	9, 12, 15	3 , 6	Section 4.8

Table 3: Expression of the Optimal Ate pairing: Miller expression power $(q^k - 1)/r$.

The results of the linear algebra allow to compute any discrete logarithm in $\mathbb{F}_{q^k}$. Since this step is much faster than the sieving and the linear algebra stages, we neglect it in the complexity analysis.

3.2 Identifying the best attacks

There is a consensus in the literature [5,31,40] that one obtains a precise estimation of the cost of exTNFS by optimizing the following equation:

$$\text{cost} = c_{\text{sieve}} \frac{2B}{\mathcal{A} \log B} \rho\left(\frac{\log_2(N_f)}{\log_2(B)}\right)^{-1} \rho\left(\frac{\log_2(N_g)}{\log_2(B)}\right)^{-1} + c_{\text{sieve}} \frac{(2B)^2}{\mathcal{A}^2 (\log B)^2 c_{\text{filter}}^2}, \quad (1)$$

where ρ is Dickman's function and $\mathcal{A}$ is the number of automorphisms of h multiplied by the number of commun number of automorphisms of f and g (which can be upper bounded by $\eta\kappa/\gcd(\eta, \kappa)$) and where c_{sieve}, $c_{\text{lin.alg}}$ and c_{filter} are constant or slowely increasig functions explained below. The validity condition is that the number of relations is larger than the cardinality of the factor base, which is as follows:

$$\frac{(2A+1)^{2\eta}}{2w} \cdot \rho\left(\frac{\log_2(N_f)}{\log_2(B)}\right) \rho\left(\frac{\log_2(N_g)}{\log_2(B)}\right) \geq \frac{2B}{\log(B)}, \quad (2)$$

where ω is the half of the number of roots of unity of h.

Comparison between two models The constants c_{sieve}, c_{filter} and $c_{\text{lin.alg}}$ are functions which increase very slowly so that they can be considered as constants up to one bit of security. In order to evaluate the reduction factor c_{filter} one can take a default value of 20 which is easily

achieved for example with the CADO-NFS software on small computations where $\log_2 p^k$ is less than 300. The reduction factor can only increase for larger computations and with new implementations, but it is hard to give an upper bound to use in security estimations. According to [5, Conjecture 1] one can take as upper bound $c_{\text{filter}} \leq \log_2 B$, and we discuss later that this safe bound gives similar results to the more realistic but unsupported value $c_{\text{sieve}} = 20$:

1. (GS model) The textbook description of NFS states that asymptotically, on a computer with infinite memory, the cost of sieving is the cost of some arithmetic operations which are negligible plus the cost of $\log_e \log_e(B)$ memory updates. Experiments of the CADO-NFS team show that the value of $c_{\text{lin.alg}}$ is the cost of $w \times \log_2 r/64$ machine word additions (replace 64 with the machine word lenght), where w is the average row weight of the matrix and r is the largest prime factor of the cardinality of the discrete logarithm group. This is in accordance with the textbook description of the block Wiedemann algorithm. Hence, Guillevic and Singh [42] took $c_{\text{filter}} = 20$ and they set $c_{\text{sieve}} = \log\log B$ and $c_{\text{lin.alg}} = 200 \log_2 r/64$.
2. (BD model) Barbulescu and Duquesne [5] took $c_{\text{filter}} = \log_2 B$ and chose the constants which best fit the cost reported by the authors of a dozen NFS records of factorization and discrete log: $c_{\text{sieve}} = 1$ and $c_{\text{lin.alg}} = 128$.

The BD model has the advantage that it is automatically scaled against the RSA key size. Indeed, our study of NFS allows to compare the cost of a given pairing to the cost of RSA-1024, we cannot directly compare a NFS computation to the security of a symmetric cryptosystem. When RSA-1024 was evaluated to 80 bits of security by the NIST recommendations, the cryptography community accepted an exchange rate between the NFS world and the symmetric cryptography. Hence, by scaling against RSA-1024 we are sure to use the same exchange rate. In the following we call normalized GS model the cost of the GS model divided by 4.
Let us recall the values of c_{filter}, c_{sieve} and $c_{\text{lin.alg}}$ used by the models of Barbulescu and Duquesne [5,31] on the one hand and Guillevic [40] on the other hand.

model	BD	normalized GS	comment
c_{filter}	$\log_2 B$	20	BD is an upper bound based on [5, Conjecture 1], GS is easily obtained by the CADO software when factoring 100 digit integers and one can hope to have at least this value in future NFS records
c_{sieve}	1	$\frac{1}{4}\log\log B \approx 1$	the BD constant is a lower bound based on the records in the literature, the GS constant is based on the textbook description of NFS. For the security table values this is actually between 1 and 1.5
$c_{\text{lin.alg}}$	128	$50\ \lceil \lvert r \rvert /64 \rceil \approx 128$	BD is an average of the records and could slowly increase with r, GS varies between 100 and 150 at 128 bits of security.

The actual value of c_{sieve} when $\eta > 1$ depends on the innovation made on the high-dimensional sieving. At the time when [5] published their model Gremy's implementation [37] had a real-life value of $c_{\text{sieve}} \approx 20$. In a recent record, McGuire and Robinson [60] reduced its value to $c_{\text{sieve}} \approx 6$. So, the model used in [5,30] remains a safe lower bound for the security of pairings whereas the model used in [40] corresponds to the state-of-the-art implementations. In this work we use the model of [5]. We repeated our key estimations in the normalized GS model and concluded that the same algorithms are the best and that the security estimation in the normalized GS model is the same as the one of the BD model or the pairing has one bit too much security. This is hence not necessary to add the GS estimations alongside the BD estimations. The BD model is at least as conservative as the GS model for all the families.
Let us see how to select f, g and h. The values of $\mathcal{A}$ and ω are a consequence of the polynomial selection and their choice is explained in [5].

Polynomial selection The choice of the polynomials f and g for NFS in $\mathbb{F}_{q^k}$ was the object of many works. When q has a polynomial form one can obtain a product $N_f N_g$ which is much smaller than in the general case. This is emphasized by putting an S, for special, before the name of each version of NFS : SNFS, STNFS or SexTNFS.

The special case Let $P \in \mathbb{Z}[x]$ and $u \in \mathbb{Z}$ be such that $q = P(u)$ and $\|P\|_\infty = O(\log(q^k))$. When k is small or prime one can use STNFS [7], i.e. h an irreducible polynomial of degree k, $f = P(x)$ and $g = x - u$, or Joux-Pierrot [46], i.e. $h = t$ (no tower), $f = P(x^k + S(x))$ and $g = x^k + S(x) - u$ where $S(x)$ is a polynomial of degree less than k. When k is large and can be written as $k = \kappa\eta$, one can use SexTNFS [50]: one chooses h to be an irreducible polynomial of degree η, $f(t, x) = P(x^\kappa + S(x) + t)$ and $g(t, x) = x^\kappa + S(x) + t - u$. When $\gcd(\kappa, \eta) = 1$ one can drop t in the definition of f and g. In a recent article Guillevic [40] proposed a method similar to the one used to factor Mersenne numbers [66].

The case of arbitrary finite fields All primes q, of polynomial or non-polynomial form, must withstand the variants of NFS for the general case. When k is small or prime one uses either TNFS [7], i.e. h is an irreducible polynomial of degree k and f and g are chosen by the "base m" method or the two algorithms of Kleinjung [52],[53], or one uses a classical variant, i.e. $h = x$ (no tower) and any of the methods of polynomial selection: GJL [6, Sec. 3.2],[59], JLSV_1 [45, Sec 3.2], JLSV_2 [45, Sec 3.1], Sarkar and Singh's algorithms A,B,C,D [68,70,69] and the Conjugation method [6, Sec 3.3]. When k is large and can be written as $k = \kappa\eta$, one uses exTNFS [50]: one selects f and g adequated for DLP computations in $\mathbb{F}_{q^\kappa}$ using the afore mentioned methods and then sets h equal to an irreducible polynomial of degree η. If $\gcd(\kappa, \eta) \neq 1$, one follows [43] and replaces the polynomials with $f(x + t)$ and $g(x + t)$.

Optimizing parameters of for NFS attacks For each construction of pairings and for each of the security levels 128, 192 and 256, we generated pairings which guarantee that the security on the curve side is greater than or equal to the required security level. The sole condition that q is prime eliminates the existence of small key sizes for many families, for example the families of embedding degree 20 or more have a field size $\log(q^k)$ greater that 6000 for 128 bits of security on the curve side. We didn't necessarily check that r is prime at this stage because one generates correct values of q and r when computing complexity and because checking the primality of r here doesn't rule out many families.
Then, for each possible choice of κ, h, f and g, we solved by SageMath scripts the optimization problem consisting in minimizing the cost in Equation (1) under the validity condition of Equation (2): For each value of $\log_2(A)$ and $\log_2(B)$ up to a precision of 0.01 we estimated experimentally N_f and N_g on a sample of 3000 pairs (a, b) chosen randomly in the sieving space. If the field side security is not sufficient, we increase the size of $\log_2 r$ and start over. We automatized the attack and the script is available on request. The complete computations took more than 1 CPU year. We summarize the results in the electronic complement available here `https://razvanbarbulescu.pages.math.cnrs.fr/Pairings/security.html`, as well as in the next section in the tables associated to each family, available at the end of the article. Our results are close to those of Guillevic [40] but the models are slightly different. We don't reproduce here the results of Fotiadis and Martindale [30,31] because they were computed by the same method as the other 150 families in this article.

3.3 An example of key size computations : RSA-1024 and MNT of embedding degree 6

RSA 1024 Kleinjung [51] made a precise estimation of the security of a 1024-bit RSA modulus and estimated it to one year on 12 million PCs with processors 2.2 GHz Athlon 64 and 2 GB of main memory. We used the polynomials proposed in Kleinjung's analysis and optimized the parameters in the two models Bd and GS. The sieving space consists of the primes up to

$56 \cdot 10^{12}$ as special-q's, each of which is made of $2^{15} \cdot 2^{16}$ pairs (a, b). It has the same cardinality and pairs (a, b) of the same size as if, in a contest without special-q, one used $\log_2 A = 38.84$. The large prime bound is taken $B = \log_2 B = 42$.
Let us now do the optimization of the parameters for the BD and GS models. The linear algebra cost is proportional to bit size length of the prime in the linear algebra : r for discrete logarithm and 2 for factoring, so we divide $c_{\text{lin.alg}}$ by 32 for BD and by $\log_2 r$ for GS. We call BD and GS the models in the literature and we call hybrid the GS model where c_{filter} equals its value in the BD model.

model	c_{filter}	c_{sieve}	$c_{\text{lin.alg}}$	$\log_2 A$	$\log_2 B$	$\log_2$ cost
BD model	$\log_2 B$	1	128/32	39.6	48.1	80.09
GS model	20	$\log(\log(B))$	200/32	40.0	49.2	82.93
hybrid	$\log_2 B$	$\log(\log(B))$	200/32	40.0	48.1	82.70
NIST recommendations						80

The GS and the hybrid models give similar results so the value of c_{filter} has a small impact on the analysis. However, the constants c_{sieve} and $c_{\text{lin.alg}}$ do not correspond to the estimation that RSA 1024 offers 80 bits of security. The parameters A and B correspond relatively well to the ones computed by Kleinjung, the parameter B being slightly larger in the models. Kleinjung didn't discuss in detail the exact choice of $\log_2 B$ so that a larger value might be slightly better. But a deeper reason might be that the models don't take into account the state-of-the-art implementation of ECM which is badly optimized to find large primes of size $\log_2 B \approx 48$.

MNT 6 at 128 bits of security Let us consider the family of Section 3.3 of the taxonomy [34] : the base field is $\mathbb{F}_q$ where q is a prime of the form $q(u) = 4u^2 + 1$, the elliptic curve order $\#E(\mathbb{F}_q)$ is $r(u) = 4u^2 - 2u + 1$ and the embedding degree equals 6, so the target of the pairing is the multiplicative group of $\mathbb{F}_{q^6}$. The polynomial form of q is important, and we must compute all the manners to write $q(u)$ as a polynomial with small coefficients. In the case of MNT 6 we take, $v = 2u$ and $P(v) = v^2 + 1$ so that $P(v) \equiv 0 \pmod{q(u)}$.
One tests in Table 4 the various algorithms and values of κ on the example of MNT-6 such that $\log_2 q = 700$. We didn't compare SexTNFS with Guillevic's polynomial selection because this is used to reduce the degree of the $q(t)$ polynomial, which is 2 for the MNT-6 family.

algorithm	κ	η	h	f	g	field security
SexTNFS	1	6	Φ_7	$x^2 + 1$	$x - u, \log_2 u = 351$	199.5
SexTNFS	2	3	$t^3 - 2t^2 - t - 1$	$x^4 + 1$	$x^2 - u, \log_2 u = 351$	141.7
SexTNFS	3	2	$t^2 + 1$	$x^6 + 1$	$x^3 - u, \log_2 u = 351$	**128.0**
SexTNFS	6	1	t	$x^{12} + 1$	$x^6 - u, \log_2 u = 351$	148.0
exTNFS base-m	1	6	Φ_7	$\sum_{i=0}^{6} f_i x^i, \log_2 u = 98$	$g_1 x - g_0, \log_2 g_i = 98$	150.5
exTNFS-Conj	2	3	$t^3 - 2t^2 - t - 1$	$x^4 + 3$	$g_1 x^2 - g_0, \log_2 g_i = 351$	141.8
exTNFS-Conj	3	2	$t^2 + 1$	$x^6 + 3$	$vx^3 - u, \log_2 g_i = 351$	128.2
exTNFS-Conj	6	1	t	$x^{12} + 3$	$g_1 x^6 - g_0, \log_2 g_i = 351$	150.0

Table 4: Security of $\mathbb{F}_{q^6}$ DLP when $\log_2 q = 700$.

We conclude that the algorithm SexTNFS with $\kappa = 3$ is the best option. For this choice we optimize the parameters A and B in Table 5.

model	c_{filter}	c_{sieve}	$c_{\text{lin.alg}}$	$\log_2 A$	$\log_2 B$	$\log_2$ cost
BD model	$\log_2 B$	1	128	31.25	70.90	128.0
GS model	20	$\log(\log(B))$	200/32	32.0	73.6	130.1
hybrid	$\log_2 B$	$\log(\log(B))$	200/32	32.0	73.6	130.0

Table 5: The SexTNFS algorithm with $\kappa = 3$ on MNT-G with $\log_2 q = 700$.

Because of the small difference in the scaling of the BD and GS models one cannot directly compare the tables computed in the two models. Given a key size the two security estimations are within a 2% error. The converse problem, given a security level, compute the key sizes is sensible on the rescaling as we show in Table 6. The GS and the hybrid models correspond to the lower and respectively the upper bound on the key sizes computed in [42], except that the bounds are slightly enlarged to take care of the uncertainty on the Monte Carlo estimation of the norms. The BD key size is 6.6% larger than the lower bound fond with the GS model. Note that the GS keys are correct in the BD model but not vice-versa.

model	c_{filter}	c_{sieve}	$c_{\text{lin.alg}}$	$\log_2 A$	$\log_2 B$	$\log_2 q$	$\log_2(q^6)$
BD model	$\log_2 B$	1	128	31.25	70.90	700	4212
GS model	20	$\log(\log(B))$	200/32	32.0	73.6	666	3948
hybrid	$\log_2 B$	$\log(\log(B))$	200/32	32.0.0	73.6	674	4008

Table 6: Computing $\log_2 q$ so that the SexTNFS algorithm with $\kappa = 3$ on MNT-6 has a cost of 128 bits of security.

3.4 Security results

We keep the model of security of Barbulescu and Duquesne [5] which is conservative in that it assumes perfect conditions for an attacker (sieving in TNFS for which no computation record is available, perfect matrix reduction in the filtering step, no memory limitation, ECM having the same performances for slightly larger smoothness bounds). The results are more precise than these obtained by forgetting the $o(1)$ term in the complexity as in [29] and [21] because we don't omit any term in Equation (1). The analysis is also more precise than that of Menezes, Sarkar and Singh [61] because we evaluate numerically the size of the norms N_f and N_g instead of using the mathematical upper bound.
In the following table we list the known families of pairings with $9 \leq k \leq 54$, which is a safety margin since the choices among BN, BLS and KSS have k between 12 and 24. The labels follow the format k, value of k, m, a two or three digits number which designs the construction number in the taxonomy [34], e.g. k9m62 denotes the family having $k = 9$ in the section 6.2 of the taxonomy, whereas k11m620 denotes the family of $k = 11$ of section 6.20 in the taxonomy.
The sizes of the Dupont-Enge-Morain and Cocks-Pinch were computed in [41] and are much slower than the other families; we don't keep them in our results. To verify the results one has to use Equation 1 and compute the best values of $\log_2 A$ and $\log_2 B$ (we provide our results and scripts on demand and we will maintain an online taxonomy together with the files which determine the security results).

k9method62	5940. 128 STNFS k=1	14450. 192 STNFS k=1	25340. 257 STNFS k=1
k9method66	5890. 128 STNFS k=1	12730. 192 STNFS k=1	29320. 256 STNFS k=1
k9method67	4764. 129 STNFS-G k=1	12570. 192 STNFS-G k=1	23260. 256 STNFS-G k=1
k9methodLZZW	5314. 128 STNFS k=1	12800. 192 STNFS k=1	21800. 256 STNFS k=1
k10method53	5306. 128 SexTNFS k=2	12250. 192 SexTNFS k=2	21450. 258 SexTNFS k=2
k10method624	4695. 128 SexTNFS k=2	9825. 192 SexTNFS k=2	22120. 256 SexTNFS k=5
k10method63	5720. 128 STNFS k=1	13630. 192 SexTNFS k=2	23080. 256 SexTNFS k=2
k10method66	5104. 142 STNFS k=1	14180. 192 STNFS k=1	30380. 256 STNFS k=1
k11method62	5412. 128 STNFS k=1	14990. 192 STNFS k=1	24860. 256 STNFS k=1
k11method620	5258. 128 STNFS k=1	10140. 192 STNFS k=1	17400. 256 STNFS k=1
k11method66	3896. 128 STNFS-G k=1	14630. 192 STNFS k=1	27700. 258 STNFS k=1
BN	5534. 128 SexTNFS k=2	13120. 192 SexTNFS k=3	25310. 256 SexTNFS k=3
k12method53	5138. 130 SexTNFS k=2	9962. 193 STNFS k=1	26590. 256 SexTNFS k=2
k12method64	6120. 134 SexTNFS k=2	12550. 192 SexTNFS k=2	24220. 256 SexTNFS k=3
k12method66	5525. 128 SexTNFS k=2	14960. 192 SexTNFS k=2	26120. 256 SexTNFS k=2
k12method67	5340. 128 STNFS k=1	14750. 192 SexTNFS k=2	20120. 256 SexTNFS k=2
k13method62	4565. 128 STNFS-G k=1	13690. 192 STNFS k=1	28830. 256 STNFS-G k=1
k13method66	4083. 154 STNFS-G k=1	8472. 192 STNFS-G k=1	18940. 256 STNFS-G k=1
k14method63	5348. 128 STNFS k=1	13330. 192 STNFS k=1	21640. 257 SexTNFS k=2
k14method66	4906. 154 STNFS k=1	11180. 192 STNFS-G k=1	27980. 256 STNFS-G k=1
k15method53	6495. 145 STNFS k=1	13520. 192 STNFS k=1	27560. 256 STNFS k=1
k15method62	8131. 175 exTNFS-Conj k=5	12210. 201 exTNFS-Conj k=5	20050. 256 exTNFS-Conj k=5
k15method620	7650. 158 STNFS k=1	12270. 192 STNFS k=1	21330. 256 STNFS k=1
k15method66	5736. 138 STNFS k=1	14150. 192 STNFS k=1	26980. 256 STNFS k=1
k15method67	9104. 188 STNFS-G k=1	12030. 206 STNFS-G k=1	23040. 256 STNFS-G k=1
k15methodDCC	5745. 139 STNFS k=1	13940. 192 STNFS-G k=1	26980. 256 STNFS-G k=1
k16method66	5608. 146 exTNFS-Conj k=4	10090. 192 exTNFS-Conj k=4	18940. 256 exTNFS-Conj k=4
k16methodKSS	5281. 142 STNFS k=1	13360. 192 STNFS k=1	23760. 257 SexTNFS-G k=2
k17method62	5152. 183 STNFS k=1	11270. 193 STNFS-G k=1	20560. 256 STNFS-G k=1
k17method66	5914. 149 STNFS-G k=1	10110. 192 STNFS-G k=1	25600. 256 STNFS-G k=1
k18method624	7929. 152 SexTNFS k=2	13330. 192 SexTNFS k=2	23650. 256 SexTNFS k=2
k18method63	8412. 155 STNFS k=1	14620. 192 STNFS k=1	16990. 287 SexTNFS k=2
k18method67	7243. 156 STNFS-G k=1	11630. 193 STNFS-G k=1	21320. 258 STNFS-G k=1
k18methodKSS	6401. 156 STNFS k=1	12180. 192 STNFS k=1	26060. 257 SexTNFS-G k=2
k19method62	5754. 145 STNFS-G k=1	11290. 194 STNFS-G k=1	20800. 256 STNFS-G k=1
k19method66	6041. 233 STNFS-G k=1	8180. 241 STNFS-G k=1	12060. 258 STNFS-G k=1
k20method64	7640. 151 SexTNFS k=2	14660. 192 SexTNFS k=2	26960. 257 SexTNFS k=2
k20method66	7013. 161 exTNFS-Conj k=4	10970. 195 exTNFS-Conj k=5	19930. 256 exTNFS-Conj k=5
k21method62	10500. 206 exTNFS-Conj k=3	15420. 244 exTNFS-Conj k=7	20570. 264 exTNFS-Conj k=7
k21method66	7135. 171 exTNFS-Conj k=3	10720. 207 exTNFS-Conj k=3	25560. 256 STNFS-G k=1
k21method67	12560. 227 exTNFS-Conj k=3	15190. 235 exTNFS-Conj k=7	19910. 273 exTNFS-Conj k=7
k22method63	10940. 161 STNFS k=1	14600. 193 STNFS k=1	27410. 257 STNFS-G k=1
k22method66	7901. 197 STNFS-G k=1	11830. 223 STNFS-G k=1	18170. 256 STNFS-G k=1
k23method62	10250. 192 STNFS-G k=1	10250. 192 STNFS-G k=1	21650. 256 STNFS-G k=1
k23method66	9614. 202 STNFS-G k=1	9614. 205 STNFS-G k=1	19290. 256 STNFS-G k=1
k24method66	7642. 167 STNFS k=1	13340. 192 STNFS k=1	24440. 256 STNFS-G k=1
k24method67	9144. 173 STNFS k=1	13750. 200 STNFS k=1	26930. 258 STNFS-G k=1
k25method62	11820. 201 exTNFS-Conj k=5	13130. 210 exTNFS-Conj k=5	20880. 259 STNFS-G k=1
k25method66	12160. 180 STNFS-G k=1	15130. 192 STNFS-G k=1	29990. 257 STNFS-G k=1
k26method624	8340. 172 SexTNFS k=2	12180. 212 STNFS k=1	18850. 256 STNFS k=1
k26method63	8346. 184 STNFS-G k=1	12440. 203 SexTNFS-G k=2	23670. 256 SexTNFS-G k=2
k26method66	7758. 209 STNFS-G k=1	11610. 234 STNFS-G k=1	16040. 257 STNFS-G k=1
k27method62	14810. 251 exTNFS-Conj k=3	17200. 266 exTNFS-Conj k=3	22250. 313 exTNFS-Conj k=3

k27method66	7638. 175 exTNFS-Conj k=3	11840. 218 exTNFS-Conj k=3	15980. 256 STNFS-G k=1
k27method67	14360. 242 exTNFS-Conj k=3	18360. 275 exTNFS-Conj k=3	24770. 322 exTNFS-Conj k=3
k27methodBLS	7697. 175 exTNFS-Conj k=3	11540. 215 exTNFS-Conj k=3	16100. 257 STNFS-G k=1
k28method53	11200. 233 STNFS k=1	16580. 247 STNFS k=1	21950. 266 STNFS k=1
k28method64	14280. 207 SexTNFS k=2	14280. 207 SexTNFS k=2	25480. 258 SexTNFS k=2
k28method66	10140. 191 exTNFS-Conj k=4	15190. 230 exTNFS-Conj k=4	20260. 261 exTNFS-Conj k=7
k29method62	8292. 232 STNFS-G k=1	15960. 245 STNFS-G k=1	18580. 257 STNFS-G k=1
k29method66	18650. 268 STNFS-G k=1	18650. 268 STNFS-G k=1	18650. 268 STNFS-G k=1
k30method53	13260. 209 STNFS k=1	19500. 236 STNFS k=1	25740. 263 STNFS k=1
k30method63	16270. 241 STNFS-G k=1	24420. 258 STNFS k=1	32580. 287 STNFS k=1
k30method66	11470. 212 exTNFS-Conj k=3	17230. 237 exTNFS-Conj k=5	22990. 270 exTNFS-Conj k=6
k30method67	16510. 231 exTNFS-Conj k=5	20900. 260 exTNFS-Conj k=5	27760. 293 exTNFS-Conj k=6
k31method62	18650. 266 STNFS-G k=1	18650. 266 STNFS-G k=1	18650. 266 STNFS-G k=1
k31method66	21780. 240 STNFS-G k=1	21780. 240 STNFS-G k=1	23900. 257 STNFS-G k=1
k32method613	14830. 227 exTNFS-Conj k=4	14870. 281 exTNFS-Conj k=4	19440. 260 exTNFS-Conj k=4
k32method66	13010. 210 exTNFS-Conj k=4	13010. 210 exTNFS-Conj k=4	19330. 257 exTNFS-Conj k=4
k48method66	13750. 290 STNFS k=1	20660. 304 STNFS k=1	27570. 320 STNFS k=1
KSS54	17060. 480 exTNFS-Conj k=2	23900. 360 STNFS-G k=1	31580. 388 STNFS-G k=1
k3MNT	4211. 128 SexTNFS k=3	9371. 192 SexTNFS k=3	16090. 256 exTNFS-Conj k=3
k4MNT	4344. 128 SexTNFS k=4	10520. 192 exTNFS-Conj k=4	19040. 256 exTNFS-Conj k=4
k6MNT	4140. 128 SexTNFS k=3	9792. 192 SexTNFS k=6	21010. 256 SexTNFS k=6

Our results are consistent with those of Guillevic [40] . At 128 bits of security on the curve side, the security on the field side is larger than or equal to 128 whenever $k \geq 13$, in all the models considered in the literature BD, GS or hybrid. Hence, the small difference between our results and the ones in [40] make no change on the key sizes of pairings with $k \geq 13$. We note for completeness that for k13method66 and k17method62 Guillevic obtains large differences between the key sizes for a general seed (Table 4 of her work) and a low weight seed (Table 5 of her work). In the case of $k = 9, 10, 11$ and 12 there are differences between the BD and the GS models, as we write in Table 6. As explained in Section 3.3, the two models are very similar, the difference is due to the security they estimate for RSA-1024.

family	$\frac{1}{2}\log_2 r$	$\log_2(p^{12})$	[5] model	[42] model
BN(method6.8)	228	5534	128	135
$BLS12$(k12method66)	153	5525	128	135
$k12$method67	128	5340	128	134
$k12$method64	128	6120	134	138

Table 8: Differences between the field security in the two models of [5] and [42] when $k = 12$.

Our results can be downloaded at:

https://razvanbarbulescu.pages.math.cnrs.fr/Pairings/Pairings.html

4 Complexity of Miller's algorithm

In this section, we search for nice parameters for the optimal Ate pairing in order to make a comparison between the most promising families at the 128, 192, and 256 bits security level. We choose the families according to two main criteria:

– the popularity of the curve in previous work, which is basically based on a smooth embedding degree multiple of 6;
– the size of the field $\mathbb{F}_{q^k}$, indeed embedding degrees that are not 0 mod 6 were not taken into account in previous work, but as the size of the finite field increase drastically for the most popular curves, we though it worth testing them. The results were interesting as according to our estimation, the most popular curves are no longer the one providing an efficient pairing.

We propose seeds for each pairing to match the security results in the previous section. We obtain the cost of Miller's loop in term of operations in $\mathbb{F}_q$ and then binary operations. Since we will obtain that the overall cost of the BLS-12 Ate pairings is 3 million 32-bit operations, we keep for the following sections only the pairings whose Miller loop is less than 3 million 32-bit operations. Similarly we keep only a short list which can beet BLS-24 for the 192 and 256 bits of security.
In Section 4.1 to Section 4.8, we study the 128 bits security level. We select one promising family by each method of construction and compare them all together in Table 15. For them we compute the cost of the final exponentiation at each level of security.
For the comfort of the reader we give all the details of the computations, but one can skip forward to the results of the Miller loop in Table 15.

4.1 Construction 6.2 from [34]

In this metafamily of curves we can construct curves whose embedding degree is odd. The curves admit a discriminant $D = -1$ (we abusively replace D in the sequel by its absolute value), so we have no twist.
The complexity of Ate pairing for construction 6.2 is $\log_2(u^2)$ doubling step, plus $HW(u^2)$ addition step and an extra doubling step for the evaluation of $\frac{l_{qQ,x^2Q}(P)}{v_{(x^2+q)Q}(P)}$.
The curves with no twist were not taken into consideration as the pairings computation cannot be improved by the denominator evaluation. We consider them in our study as they are quite resistant to the NFS attack. As a consequence, the size of $\mathbb{F}_q$ is smaller for curves without twist and the number of doubling step for the Miller algorithm is also smaller.
To our knowledge, there is no reference in the literature to pairing computations without twists. We computed new formulae and we obtain the arithmetic cost of each step in Table 9.

Operation	Complexity affine	Complexity projective	Complexity Jacobian
Doubling step	$2M_k + S_k + I_k$	$3kM_q + 12M_k + 7S_k$	$3kM_q + 10M_k + 8S_k$
Addition step	$5M_k + 2S_k + I_k$	$3kM_q + 16M_k + 2S_k$	$3kM_q + 19M_k + 14S_k$

Table 9: Complexity of Miller's steps without twist

We use the estimation $M_k = S_k$ and find that the doubling step in projective coordinates has a cost of $3kM_q + 19M_k$. Compare this to that in Jacobian coordinates which is $3kM_q + 18M_k$. For the addition step, the difference between the two types of coordinates is more important : in projective coordinates we obtain $3kM_q + 18M_k$ and in Jacobian ones we get $3kM_q + 33M_k$. Let α denotes the length of Miller loop and $HW(\alpha)$ be the Hamming Weight of α. The complexity of the pairing evaluation without twist is more efficient for projective coordinates when compared with Jacobian as long as $15HW(\alpha) \geq \log_2(\alpha)$. As our goal is to give a first estimation of the pairing complexity, we do not search especially for parameters with very small Hamming weight. Note that the affine coordinates could be more interesting than the projective ones if the complexity of the inversion in $\mathbb{F}_{q^k}$ is smaller than $20M_k$. This coarse estimation is obtained by considering that $M_k = S_k$ and $kM_q = M_k$. The expected gain is not important enough, so we don't continue with a precise estimation in this case.
The curves of embedding degree 9 are the champion among the curves of construction 6.2 without twists. Yet, they are no match for the curves admitting twists in following constructions.

4.2 Construction 6.3 from [34]

Using this construction, we obtain elliptic curve having an embedding degree $k = 2k'$, for k' an odd number. Those curves have a discriminant $D = 1$, they admit a twist of degree 2.
The optimal Ate pairing for curves constructed using method 6.3 consists in one Miller's algorithm indexed over x^2, plus an extra line evaluation.
The Table 10 presents the value that we find by a quick research and using very large estimation for the cost of arithmetic in the tower field. We used the estimation cost from Table 2 as we are working on elliptic curve with discriminant 1 and quadratic twist.
The smallest number of iterations for Miller's algorithm could be reached for the curve with $k = 38$, but unfortunately, in practice, we do not find a value of u that makes q and r prime below 15 bits.
The smallest size for $\mathbb{F}_q$ is theoretically obtained for the curve with embedding degree 26, 34 and 46. Together with the theoretically smallest number of iterations during the Miller algorithm. In practice, the less expensive Miller's algorithm corresponds to $k = 14$. For this value we also have the smallest finite field $\mathbb{F}_q$. As a consequence, the best choice for the method 6.3 using a quadratic twist at the 128 bits of security should be the curve with $k = 14$.

4.3 Construction 6.4 from [34]

In this metafamily of curves, we construct curves with embedding degrees $4k'$ where k' is an odd integer. The discriminant is $D = 1$, consequently, curves in this family admit a twist of degree 4.
The optimal Ate pairing for curves constructed using method 6.4 is composed by one Miller's algorithm indexed over x, plus an extra line evaluation. The Table 10 presents some examples of values for u that minimize the number of addition steps during Miller's algorithm.
We compare the curves with approximately 10 000 M_q ($k = 12, 20, 28$) and the curve with the smallest field $\mathbb{F}_q$ ($k = 44$). On a 32 bits architecture, it seems that the curves constructed by method 6.4 with $k = 28$ provides the most efficient pairing, on a 64 bits architecture, it should be the curve with $k = 20$. Of course, those results highly depends on the architecture and the implementation.

4.4 Construction 6.6 from [34]

In this metafamily of curves, also called BLS, we can construct curves with discriminant $D = 3$. Hence, in this case the elliptic curves can admit a twist of degree 3 or 6. The method of construction depends on the residue of k modulo 6, and we studied all the families from $k = 9$ to $k = 53$, all being possible except those for which 18 divides k, i.e. 18, 36 and 54.

Curves admitting a twist of degree 6 When $k = 0 \bmod 6$, then the elliptic curve admits a twist of degree 6. The corresponding embedding degrees are $k \in \{12$ (i.e. BLS12), 24 (i.e. BLS24), 30, 36, 42, 48 $\}$.
The smallest number of operation over $\mathbb{F}_q$ is obtained for $k = 12$, but the smallest field is obtained for $k = 24$.
In order to compare those two curves, we have to estimate the complexity of the Miller algorithm in terms of machine word. The Table 12 presents our estimation. We consider that a multiplication over $\mathbb{F}_q$ is computed using the schoolbook multiplication.
According to our estimation, the optimal Ate pairing seems to be more efficient on BLS24 than on BLS12 curves.

Curves admitting a twist of degree 3 Among the elliptic curves constructed by method 6.6, those for which $k = 3 \bmod 6$ admit a twist of degree 3. The expression of the optimal Ate pairing depends on the embedding degree. For each embedding degree $k \in \{15, 21, 27, 33, 39, 45, 51\}$, we obtain a different short vector that should be used in order to compute the pairing. The expression of the pairing follows a common pattern for $k \in \{15, 33, 51\}$, respectively for $k \in \{27, 45\}$; and for $k \in \{21, 39\}$.
For $k \in \{15, 33, 51\}$ using the construction 6.6, we obtain the same pattern for a short vector: $[x, -1, 0, \ldots 0, -1, 0, \ldots, 0]$.
We give here the definition of an optimal Ate pairing for $k = 15$.
We choose $[x, -1, 0, 0, 0, 0, -1, 0, \ldots, 0]$ as short vector. The expression of the optimal Ate pairing using this vector is the following:
$OptAte_{k15_6.6d3} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$,
$(P, Q) \to \left((\frac{f_{x,Q}}{v_Q^{q+q^6}} \frac{l_{s_1Q,xQ}}{v_{s_0Q}} \frac{l_{s_2Q,-qQ}}{v_{s_1Q}})(P) \right)^{\frac{q^k-1}{r}}$, where $s_0 = x - q - q^6$, $s_1 = -q - q^6$ and $s_2 = -q^6$.
When using a twist of degree 3, the vertical line does not vanish during the final exponentiation. We can however simplify the pairing expression. Zhang and Lin in [81] proposes the latest record for the computation of pairings over curves with a twist of degree 3. They barely improve the result of [20] but the method is very helpful for the simplification of the optimal Ate pairing in our case. We use Zhang and Lin formulas for the complexity of Miller's algorithm's step 2.
Applying the method developed by Zhang and Lin in [81], we can make the following transformation $\frac{1}{(v_Q)}(P) = \frac{X_Q^2 + X_Q Z_Q x_p + x_q^2}{Z_Q^2}$.
Indeed, using the method developed by Zhang and Lin in [81], we can transform the fraction $\frac{l_{s_1Q,xQ}}{v_{s_0Q}}$ into

$$X_{s_0Q}^2 - Z_{s_1Q} Z_{xQ} (Z_{s_1Q} X_{xQ} - X_{s_1Q} Z_{xQ})^2 (Z_{s_1Q} Y_{xQ} - Y_{s_1Q} Z_{xQ})(Y_{s_0Q} - Z_{s_0Q} y_P) +$$
$$X_{s_0Q} Z_{s_0Q} x_P + Z_{s_0Q}^2 x_q^2$$

which correspond to an extra addition step $s_0Q = s_1Q + xQ$. We can apply the same method to the other fraction $\frac{l_{s_2Q,-qQ}}{v_{s_1Q}}$. The Miller algorithm output the point xQ. We remark that $s_1Q = s_2Q + (-Q^q)$, thus the evaluation of $\frac{l_{s_2Q,-qQ}}{v_{s_1Q}}$ correspond to the addition step between s_2Q and $-Q^q$. We also can notice that $s_0Q = s_1Q + xQ$, we then obtain that $\frac{l_{s_1Q,xQ}}{v_{s_0Q}}$ correspond to the addition step between s_1Q and xQ the output of Miller's algorithm.In order to perform these computations, we have to precompute the points $s_2Q = -Q^{q^6}$, $s_1Q = -Q^q + Q^{q^6}$ and $s_0Q = xQ - Q^q + Q^{q^6}$. Those computations correspond to two Frobenius Q^q and Q^{q^6}. We follow the example of [5] and the coarse estimation that a Frobenius evaluation cost $(k-1)M_q$.
We want to simplify the evaluation of $\frac{1}{(v_Q)^{q+q^6}}$. The power $q + q^6$ could be split into two Frobenius evaluation. We will modify the expression of $\frac{1}{(v_Q)}$ by the following way:

$$\begin{aligned}
\frac{1}{(v_Q)}(P) &= \frac{1}{x_Q - x_P} \text{ we begin with affine coordinates} \\
&= \frac{(y_Q^2 - y_q^2)}{(x_Q - x_P)(y_Q^2 - y_q^2)}, \\
&= \frac{x_Q^2 + x_Q x_P + x_q^2}{y_Q^2 - y_q^2}.
\end{aligned}$$

Using a twist of degree 3, we have that $y_Q^2 - y_q^2$ belongs to $\mathbb{F}_{q^{k/d}}$ and as a consequence will vanish during the final exponentiation.
In [81], the authors made the assumption that affine coordinates should be more efficient than projective one as long as $I_k \leq 5.6M_k$. In order to be the more general, we will consider only the

projective coordinates. We than transform the affine expression into the following projective one:

$$\frac{1}{(v_Q)}(P) = \frac{X_Q^2 + X_Q Z_Q x_p + x_q^2}{Z_Q^2}.$$

When using a twist, the coordinates Z_Q belongs to $\mathbb{F}_{q^{k/d}}$.
As a consequence, the evaluation of $\frac{1}{(v_Q)}$ is composed by $S_q + kM_q + S_{k/d} + M_{k/d}$ operations. We need two Frobenius maps (one by q and one by q^6) plus M_k in order to compute $\frac{1}{(v_Q)^{q+q^6}}$. Finally the total complexity of $(\frac{f_{x,Q}}{v_Q^{q+q^6}} \frac{l_{s_1Q,xQ}}{v_{s_0Q}} \frac{l_{s_2Q,-qQ}}{v_{s_1Q}})(P)$ is the computation of Miller's algorithm plus $(5k-4)M_q + S_q + S_{k/d} + M_{k/d} + 2M\mathbf{A} + 2M_k$. We present in Table 11 the estimation of the Miller algorithm when $k \in \{15, 33, 51\}$.
For $k \in \{27, 45\}$ we obtain a short vector on the pattern $[x, 0, \ldots 0, 1, 0, \ldots, 0]$. The optimal Ate pairing expression is then $\left(f_{x,Q} \frac{l_{q^{10}Q,xQ}}{v_{(x+q^{10}Q)}}(P)\right)^{\frac{q^k-1}{r}}$. An alternative family for the BLS 27 family was proposed by Zhang and Lin [81]. They used a substitution of x by $-1/x$. The optimal Ate pairing expression is simplified into $(f_{x,Q})^{\frac{q^k-1}{r}}$. Another advantage to the Zhang and Lin family for BSL27 is the existence of x such that q and r are both prime.
For $k = 45$, the fraction is $\frac{l_{q^{16}Q,xQ}}{v_{(x+q^{16}Q)}}$.
As a consequence, for $k \in \{27, 45\}$ the pairing complexity is one Miller execution, plus one addition step.
For $k = 21$, we obtain this short vector $[0, 0, 0, 0, 0, 0, x^2, -x, 1, 0, 0, 0]$ and for $k = 39$ this one $[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, x^2, -x, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]$.
We obtain the following expressions for the pairings $\left(\frac{f_{x^2,Q}^{q^6}}{f_{x,Q}^{q^7} v_{xQ}^{q^7}} \frac{l_{s_7Q,x^2Q}}{v_{s_6Q}} \frac{l_{s_8Q,-xqQ}}{v_{s_7Q}} \frac{v_Q}{v_{s_8Q}}(P)\right)^{\frac{q^k-1}{r}}$,
where $s_6 = x^2q^6 - xq^7 + q^8$, $s_7 = -xq^7 + q^8$ and $s_8 = q^8$ and $\left(\frac{f_{x^2,Q}^{q^{12}}}{f_{x,Q}^{q^{13}} v_{xQ}^{q^{13}}} \frac{l_{s_{13}Q,x^2Q}}{v_{s_{12}Q}} \frac{l_{s_{14}Q,-xqQ}}{v_{s_{13}Q}} \frac{v_Q}{v_{s_{14}Q}}\right)^{\frac{q^k-1}{r}}$,
where $s_{12} = x^2q^{12} - xq^{13} + q^{14}$, $s_{13} = -xq^{13} + q^{14}$ and $s_{14} = q^{14}$.
The pairing computation consists in one Miller execution as its result, $f_{x,Q}$, is an intermediate step of the computation of $f_{x^2,Q}$. The point xQ can also be saved during the execution of $f_{x^2,Q}$. The output is the point x^2Q. We must perform 6 Frobenius. The computation of $\frac{l_{s_{13}Q,x^2Q}}{v_{s_{12}Q}} \frac{l_{s_{14}Q,-xQ}}{v_{s_{13}Q}}$ are two extra addition steps. The denominators $v_{s_{13}Q}$ and $v_{s_{14}Q}$ cost $2(S_q + kM_q + S_{k/d} + M_{k/d})$. The complexity of the pairing computation for $k = 21$ and $k = 39$ is then one Miller execution $f_{x^2,Q}$ plus the extra computations $26(k-1)M_q + 2M\mathbf{A} + 2(S_q + kM_q + S_{k/d} + M_{k/d}) + 5M_k + I_k$. The Table 11 presents our results. The best candidates among those curves are for $k = 15$ and $k = 27$.

Curves admitting a twist of degree 2 The curves constructed using method 6.6 admits a twist of degree 2, when $k \bmod 6 \in \{2, 4\}$. This means that $k \in \{14, 16, 20, 22, 26, 28, 32, 34, 38, 40, 44, 46, 50, 52\}$.
The optimal Ate pairing expression depends on the value of $k \bmod 6$. For every $k = 2 \bmod 6$ we find the same short vector: $[x^2, x, 1, 0, \ldots 0]$. The expression of the optimal Ate pairing is then $\left(f_{x^2,Q} f_{x,Q}^q l_{s_1Q,x^2Q} l_{s_2Q,xqQ}\right)^{\frac{q^k-1}{r}}$, where $s_0 = x^2 + xq + q^2$, $s_1 = -xq + q^2$ and $s_2 = q^2$.
As the results xQ and $f_{x,Q}$ are computed during the computation of $f_{x^2,Q}$ we count only one Miller evaluation. Two line evaluations plus 3 Frobenius and $3M_k$ are also necessary.

Its complexity is equal to $\log_2(u^2)$ doubling steps, plus $HW(u^2)$ addition steps and an extra doubling step for the evaluation of $l_{qQ,x^2Q}(P)$. As we do not need the coordinates of the point $(x^2+q)Q$, this line evaluation (Le) is cheaper than a full doubling step [5][4]. We use the projective coordinates, which are better than the affine ones at 128 bits of security [20,81].
The Table 11 presents the cost of the Miller execution.
When $k = 4 \bmod 6$, one short vector is $[x^2, 0, \ldots, 0, -x, 0, \ldots, 0, 1, 0, \ldots, 0]$. For instance, for $k = 16$, the optimal Ate pairing is then
$\left(\frac{f_{x^2,Q}}{f_{x,Q}^{q^3}} l_{s_1Q,x^2Q} l_{s_2Q,-xq^3Q}\right)^{\frac{q^k-1}{r}}$, where $s_0 = x^2 + xq^3 + q^6$, $s_1 = -xq^3 + q^6$ and $s_2 = q^6$. The cost is one Miller execution, plus 3 Frobenius, two line evaluations, $3M_k$ and one inversion over $\mathbb{F}_{q^k}$.
Note that $D = 1$ and the equation of the elliptic curve is $y^2 = x^3 + ax$. We use the formulas from [20].
The Ate pairing computation is more efficient. Indeed, it is composed of one execution of the Miller algorithm, which has $\log_2(u^2)$ iterations using the denominator elimination. The vertical line $v_{(x^2+q)Q}(P)$ belongs to $\mathbb{F}_{q^{k/2}}$ and is eliminated by the final exponentiation. The Ate pairing expression is simplified into:
$\left(f_{x^2,Q}(P) \times l_{qQ,x^2Q}(P)\right)^{\frac{q^k-1}{r}}$.
Its complexity is equal to $\log_2(u^2)$ doubling steps, plus $HW(u^2)$ addition steps and an extra doubling step for the evaluation of $l_{qQ,x^2Q}(P)$ which is also cheaper than a full doubling step. We use the projective coordinates, which are better than the affine ones at 128 bits of security [20,81].

Curves without twists The remaining elliptic curves ($k = 1$ or $5 \bmod 6$) do not admit twists. As we have seen for construction 6.2, even if the theoretical dimension of $\mathbb{F}_{q^k}$ is smaller for prime embedding degree than for not prime embedding degrees, the lack of denominator elimination is a heavy drawback.
The complexity of the optimal Ate pairing computation is one Miller's algorithm execution, two extra addition steps, two Frobenius, hence a total of $5M_k + I_k$ operations.
Comparison among the method 6.6 (BLS) pairings. The curve BLS27 in the version of Zhang and Lin provides the smallest field $\mathbb{F}_q$ and the smallest number of operation over $\mathbb{F}_q$. This curve seems to provide the most efficient choice when considering the Miller loop among the BLS families. We analyse the final exponentiation in Section 5. The curves BLS 24 seems to provide the second most efficient Miller loop. Considering that, the BLS 24 curves have a degree 6 twist and that $\log_2(q_{24}^k) = 7656$ (when $\log_2(q_{27}^k) = 8058$), the comparison with the final exponentiation will decide between this two curves. Potentially, the BLS 15 curves could also be a competitor if a nice arithmetic over $\mathbb{F}_{q^5}$ can be deployed. Indeed, if we compare $\log_2(q_{15}^k) = 5745$ and $\log_2(q_{24}^k) = 7656$, which is roughly the size of the exponent for the final exponentiation, the BLS15 curve provide smaller field but the BLS24 curve can be implemented using the compressed squarings when no practical optimization are available in the literature for $k = 15$. As a conclusion, a precise implementation and analysis is necessary, in order to choose one between those three families.

4.5 Construction 6.7 from [34]

In this metafamily, we can construct curves with discriminant $D = 2$. They admit a twist of degree 2 if k is even, and no twist otherwise.

Curves having a twist of degree 2 The optimal Ate pairing is different for $k = 12$, $k = 24$ and respectively $k \in \{18, 30\}$. The formulas are presented in Table 3. Table 10 presents the complexity of its implementation. The curves k12m67 and k24m67 are the most promising for this family.

[4] We count $5M_e$ in the evaluation of Le instead of $4M_e$ as presented in [5] because when we wrote down the equation we do not see how to save one more M_e

Curves without twists The optimal Ate pairing is different for $k = 15$ and $k \in \{9, 21, 27\}$. For $k = 15$, the shortest vector found is $[x^4 - 1, 1, 0, -1, 1, -1, 0, 1]$, the cost of the optimal Ate pairing in this case is the evaluation of $f_{x^4-1,Q}$, plus 6 addition steps, hence a total of $10M_k + I_k$.

For $k \in \{9, 21, 27\}$, it is $\left(f_{x^4,Q}\frac{l_{q^5Q,x^4Q}}{v_{[x^4+q^5]Q}}\right)^{\frac{q^k-1}{r}}$.

For $k = 21$, there are very few possible values for u, so that we could not provide a realistic example of such pairing,

Best candidate for method 6.7 The cost of Miller's loop for the curves without twists is much more expensive than the cost for curve with a quadratic twist. Among the curves with quadratic twists, the curves with $k = 12$ and $k = 24$ are the most promising. With $k = 12$ we have the least number of operation over $\mathbb{F}_q$, with $k = 24$ the smallest field $\mathbb{F}_q$. According to our estimation, the most efficient pairing for curves constructed with method 6.7 should be implemented over the curve with $k = 12$.

4.6 Construction 6.20, 6.24 and "+" from [34]

We denote by "+" the construction described in [34] that relies on the application of Theorem 6.19 [34]. The method is to use one construction among 6.2, 6.3, 6.7, 6.20 or 6.24 and made the substitution $x^2 \to \alpha x^2$ in the definition of q and r, where α is a square free positive integer. The best choices for α are described in the Algorithm for Generating Variable-Discriminant Families [34]. The "+" doesn't change the security (and hence doesn't change the key sizes) because we obtain the same values of k, $\log_2 q$ and polynomials in the SexNFS attacks. Indeed, if the fastest SexTNFS attack against a family uses two polynomials f and g, one could use either the same polynomials or $f(\alpha x^2)$ and $g(\alpha x^2)$ for the "+" family. However, the degree of f and g is "too high" for all the families tested, so an attacker is bound to continue to use f and g.
For example , using the "+" method, we generate values of u such that $\log_2(u) = 13$ for $k = 11$ and construction 6.20, but for 128 bits of security u should be at least 20 bits. One can use our results and try to generate curves with nice discriminant. It is very important to remark that using the construction "+", we can construct elliptic curve with any discriminant. For instance, in the construction 6.2, when $k = 3 \bmod 6$, we cannot use any twist, but with construction "6.2+", we can generate curves with discriminant $D = 3$ and then use twists in order to improve the computation. By the same way, when $k = 0 \bmod 6$, the construction 6.2 allows a quadratic twist, while the construction "6.2+" allows a sextic twist.
Using construction 6.20 and 6.24, we obtain elliptic curves with discriminant $D = 1$. As a consequence, if k is even, we have a quadratic twist, otherwise we do not have a twist. For some embedding degrees, $q(x)$ is reducible so we had to apply the "+" construction.
The only drawback of the "+" method is that instead of searching for parameters u of a given bit size b we search for parameters y_0 of approximately $b/2$ bits. This gives less choices and we could not find parameters of low NAF weight for the constructions 6.20+ and 6.24+. We leave it as an open problem the generation of nice parameters and curves using the "+" method.

4.7 KSS families from [34]

The KSS families of elliptic curve were introduced by [48]. It is a promising complete family for specific values of k. They are defined for $k = 16, 18, 32, 36, 40$ in [48]. Scott and Guillevic [74] found a similar family with $k = 54$.
The KSS16 and KSS18 were already studied in the literature, we confirm the results from [5].
For $k = 32$, an expression of the optimal Ate pairing is $f_{x,Q}f_{-3,Q}^q f_{2,Q}^{q^8} l_{s_1Q,xQ} l_{2q^8Q,-3Q}$, with $s_1 = -3q + 2q^8$. This is almost the same expression for KSS36 curves, the difference is that the

power of q is 7 and not 8. For both KSS32 and KSS36 curves, we search for a value u such that the most significant bits are both 1, this will guarantee that the computation of $3Q$ is the first addition step during the computation of $f_{x,Q}$. As a consequence the cost of this optimal Ate pairing is one Miller execution $f_{x,Q}$ plus $3\pi_q + 2\mathbf{L} + 4M_k + I_k$.
For $k = 40$, $f_{x,Q} f_{2,Q}^{q^{11}} l_{s_1 Q, xQ} l_{2q^{11}Q,-Q}$, with $s_1 = -q + 2q^{11}$. The cost is $f_{x,Q}$ plus $2\pi_q + 2\mathbf{L} + 3M_k$.
For $k = 54$, $f_{x,Q}^{q^9+1} l_{q^9 xQ + q^{10}Q, xQ} l_{q^{10}Q, q^9 xQ}$ [74].

4.8 Other families

The article [34] presents a non exhaustive list of pairing-friendly elliptic curve constructions at the beginning of 2010.
The MNT curves [62] are ordinary curves with embedding degree $k = 3, 4, 6$. In [65,72,55], some examples of MNT curves are given. These parameters are more rare than for the complete families and the algorithms to compute them are more costly, so it is beyond the scope of this article to propose numerical values of u. A non exhaustive list is available in[58]. In our work, we estimate the cost of Miller's loop for this curves, but when considering Table 12, the MNT family is not at all competitive.
There were other constructions like [24,57] not included in [34]. In 2010, the ρ value was important when considering the efficiency of pairings. The curves constructed in [24] have embedding degree already included in [34] but with larger ρ. It could be a reason why the results from [24] were not included in [34]. However, the curve with embedding degree 15 in [24] resists better the Kim-Barbulescu attack and we choose to evaluate them in our study. In [24], other families are constructed with embedding degree $k = 12, 13, 14, 24, 48$. They do not provided efficient pairings, either because of the lack of discriminant $D = 3$ ($k = 13, 14$) or because the Kim-Barbulescu attack is very efficient and the required bit sizes make the pairing less efficient than others families ($k = 12, 24, 48$).
The k=9 family from [57] and the k=15 family from [24] were studied in [32], where Fouotsa et al. evaluate the cost the optimal Ate pairing computation for curves with odd embedding degree. The expression of the optimal pairing for this family is nice: $(f_{x,Q})^{\frac{q^k-1}{r}}$. It is the same expression for the family with embedding degree 9 studied by Lin et al. in [57]. Their results were that the $k = 9$ family is a little bit more expensive than the BN family.
We report in Table 14 the estimation of the Miller loop for those families at the 128 bits security level. We **A**, the results for BN curves. According to our new security evaluation, the results from [5] do not provide exactly the 128 bits security, a nice candidate could be $u = 1 + 2^3 + 2^{13} + 2^{14} + 2^{32}$ but the complexity of pairing over BN curves is less efficient than others and we keep the same results as [5].
Between those three curves, the construction from [24] with $k = 15$ is the more efficient when considering the Miller loop. We provide in Section 5 the expression of the final exponentiation in order to decide between those two families. The BN family is no longer a good choice for pairing-based cryptography.

4.9 Complexity of the Miller's algorithm at 192 bits security level

We only provide here our most efficient curves for each construction.
We select one promising family by method of construction and compare them all together in Table 15.
It seems that the curve with $k = 27$ and construction 6.6 version Zhang Lin could provide the most efficient Miller's algorithm at the 192 bits security level. Other good candidates could be BLS 15, BLS 24 $k = 28$ construction 6.4 and DCC 15. The final exponentiation could shuffle this ranking. In Section 5 we compare the cost of the final exponentiation in order to determine which curve will provide the most efficient optimal Ate pairing.

4.10 Miller's complexity at 256 bit security level

We choose to give the estimation of the pairing computation for the curves such that $\log_2(q^k)$ is not greater then 15 000 and of course to the curves that provide efficient pairing implementation at 128 and 192 bits security level.

The curves providing $\log_2(q^k) \leq 15000$, are curves without twist and/or expensive pairing computation. We found out that even if the extension field $\mathbb{F}_{q^k}$ is not very large, the estimation cost for the Miller loop (see Table 15) is much more expensive than curves admitting twists reported in Table 15.

According to Table 15, the most efficient Miller's loop would be for the curves $k = 28$ construction 6.4 in [34], BLS15 and BLS27. Those curves correspond to the families such that $\log(q)$ is smaller than 1 000 bits.

5 The Computation of the final exponentiation

The computation of Tate pairing and its variants, e.g. Ate, require two steps : Miller's loop (treated in Sections 4, 4.9, and 4.10) and the final exponentiation. None of the two steps is negligible : whereas in the earliest implementations of pairings Miller's loop was more expensive, the final exponentiation has become a significant component of the global computation. For example the family BLS-27 which is the champion at 256 bits of security is an exception where the final exponentiation dominates. We do the first analysis of the final exponentiation as previous results in the literature [39] only consider Miller's loop.

Thanks to the cyclotomic polynomial, the final exponentiation can be broken down into two components as follows:

$$\frac{q^k - 1}{r} = \frac{q^k - 1}{\phi_k(q)} \times \frac{\phi_k(q)}{r}$$

where k is the embedding degree.

In this work, we are only interested in the computation of the second factor, called the hard part, which dominates the computations of the final exponentiation. The computation of the easy part, not treated, requires merely several Frobenius computations (2 if k is even), several multiplications and an inversion in $\mathbb{F}_{q^k}$.

In Section 4 we explained why we can make a short list of the complete computations based only on the analysis of Miller's loop. Hence we have a preliminary short list consistng only of pairings of embedding degree $k = 9$, 15, 12, 16; 20; 24 and 28 for the 128 bits security level. For the security levels 192 and 256, we use the same method presented below, we have just to change the parameter u.

Throughout this section, d denotes the hard part of the final exponentiation, i.e, $d = \frac{\phi_k(q)}{r}$ and d' denotes a multiple of d with r not dividing d'.

We keep the notations M_q, S_q, I_q for the cost of the multiplication, of the squaring and of the inversion in $\mathbb{F}_q$ and similarly M_k, S_k and I_k for the operations in $\mathbb{F}_{q^k}$ as they were introduced in Section 2.1. When it is clear from the context we drop the k index and write M, S and I for M_k, S_k and I_k. We add the notations E_u for an exponentiation by the parameter u and F_k for the cost of a Frobenius map in $\mathbb{F}_{q^k}$.

As we said in the introduction of this work, we computed the final exponentiation (easy part+hard part) of the Optimal Ate pairing defined in several elliptic curves of different embedding degrees. Since we can not give all computation details in this paper version, we invite the reader to check the complete version available on Eprint [8].

In the current version, we chose to give the details about computing the final exponentiation of the Optimal Ate pairing on elliptic curves of embedding degree $k = 12$, 18, and 27

5.1 The case of $k = 12$

We showed in Section 4 that for computing Miller loops in the case of elliptic curves of embedding degree $k = 12$, it is better to consider BLS12 than BN curves. In this paragraph, we compare the cost of the final exponentiation of Optimal Ate pairing in both curves. Recall that

$$\frac{q^{12}-1}{r} = (q^6-1) \times (q^2+1) \times \frac{q^4+q^2+1}{r}.$$

The computation of the first part of the final exponentiation, i.e: the result of Miller loop raised to power $(q^6-1) \times (q^2+1)$, has almost the same cost for the two families (2 $q-$Frobenius, 2 multiplications and one inversion in $\mathbb{F}_{q^k}$ a finite field of 5535 bits for BN curves and respectively 5532 bits for BLS curves).
We present now the cost of computing the second part.

BN curves: We briefly present the BN elliptic curve [10] which is defined over $\mathbb{F}_q$ by $E: y^2 = x^3 + b$, where $b \neq 0$ is neither a square nor a cube and by a parameter u such that

$$r = 36u^4 + 36u^3 + 18u^2 + 6u + 1 \quad \text{and} \quad q = 36u^4 + 36u^3 + 24u^2 + 6u + 1.$$

The parameter u is chosen such that both q and r are prime numbers, we consider the parameter suggested in [5]: $u = 2^{114} + 2^{101} - 2^{14} - 1$.
From the given expressions of q and r, the hard part of the final exponentiation can be written as a function of u:

$$\frac{q^4-q^2+1}{r} = \Lambda_0 + \Lambda_1 q + \Lambda_2 q^2 + \Lambda_3 q^3 \quad \text{with} \quad \begin{cases} \Lambda_0 = -36u^3 - 30u^2 - 18u - 2, \\ \Lambda_1 = -36u^3 - 18u^2 - 12u + 1, \\ \Lambda_2 = 6u^2 + 1, \\ \Lambda_3 = 1. \end{cases}$$

There are many efficient methods for computing the hard part of the final exponentiation presented in [73], [22], [35] and in [26]. In this paragraph we present our new development of the multiple of this part presented by Fuentes et *al.* in [35], which makes the computation of the part in question more efficient (we know that an exponent of a pairing is a pairing). So we give the following presentation:

$$\begin{aligned} 2u\left(6u^2+3u+1\right)\frac{q^4(u)+q^2(u)+1}{r(u)} &= \left(12u^2(u+1) - 6u^2 + 4u - 1\right)q^3 + \left(12u^2(u+1) - 6u^2 + 6u\right)q^2 \\ &\quad + \left(12u^2(u+1) - 6u^2 + 4u\right)q + \left(12u^2(u+1) + 6u + 1\right), \\ &= \Lambda'_3 q^3 + \Lambda'_2 q^2 + \Lambda'_1 q + \Lambda'_0, \end{aligned}$$

$$\text{with,} \quad \begin{cases} \Lambda'_0 = (12u^2(u+1) + 6u) + 1 = c + 1, \\ \Lambda'_1 = (\alpha_2 - 2u), \\ \Lambda'_2 = c - 6u^2, \\ \Lambda'_3 = \alpha_1 - 1. \end{cases}$$

Since the parameter u is odd, an exponentiation by $u+1$ is more efficient than by u since $WH(u+1) < WH(u)$. Therefore, our algorithm for computing the hard part of the final exponentiation, is more efficient than the methods presented in [26] and [5]. Our algorithm requires $2E_u + E_{u+1} + 9M_{12} + 3S_{12} + 3F_{12}$. The overall cost of the final exponentiation is $3E_u + 10M_{12} + 3S_{12} + 5F_{12}$. In term of complexity in $\mathbb{F}_q$, our method for computing the final exponentiation requires $7381M + I$ when we use the cyclotomic squaring and $5598M + 4I$ in the case of considering the compressed squaring in the cyclotomic subgroup.

BLS12 curves: BLS12 [9] are defined over $\mathbb{F}_q$ by $E : y^2 = x^3 + b$ and by a parameter $u \in \mathbb{Z}$ such that:

$$\begin{cases} q = (u-1)^2(u^4 - u^2 + 1)/3 + u, \\ r = u^4 - u^2 + 1, \\ t = u + 1. \end{cases}$$

For computing the hard part of the final exponentiation, we refer to the algorithm presented in [36]. For the 128 security level, we consider the parameter $u = -2^{77} + 2^{50} + 2^{33}$. Then, in terms of complexity in $\mathbb{F}_q$, the final exponentiation requires $8151M + I$ when we use the cyclotomic squaring and $6188M + 6I$ in the case of considering the compressed squaring in the cyclotomic subgroup.
For the 192 security level, we consider the parameter $u = -2^{207}+2^{206}+2^{105}+2^{11}+2^7+2^6+2^2+2$. Then, in terms of complexity in $\mathbb{F}_q$, the final exponentiation requires $21201M + I$ when we use the cyclotomic squaring and $15500M + 6I$ in the case of considering the compressed squaring in the cyclotomic subgroup.

5.2 The case of $k = 18$

In this paragraph, we give the cost of computing the final exponentiation of the Optimal Ate pairing on elliptic curves of embedding degree $k = 18$.
For the complexity of computing the final exponentiation for the 128-bit security level we consider the parameter u presented in [5] $u = 2^{44} + 2^{22} - 2^9 + 2^6$ requires 20141M+I when considering the cyclotomic squaring and 17831M+8I when considering the compressed squaring.
For the 192 security level, we consider also the parameter u proposed in [5] $u = 2^6 - 2^{26} - 2^{31} - 2^{85}$. With this parameter, the computation of the final exponentiation requires 30473M+I when considering the cyclotomic squaring and 24719M+8I when considering the compressed squaring.
For the 256 security level, we consider the parameter u proposed in 4.10, $u = 2 - 2^3 - 2^7 - 2^{12} + 2^{15} + 2^{16} + 2^{20} + 2^{174}$. The complexity of the final exponentiation when using this parameter requires 55925M+I when considering the cyclotomic squaring and 42695M+8I when considering the compressed squaring (41687M+8I in the case of using the parameter u proposed in [5] $u = 2^{186} + 2^{75} - 2^{22} + 2^4$).

5.3 The case of $k = 24$

BLS curves of embedding degree 24 are important candidates for computing Optimal Ate pairing for both of the 128 and 192 security levels [5]. Recall that BLS24 curves are families of elliptic curves defined over $\mathbb{F}_q$ by the parametrization:

$$\begin{cases} q = (u-1)^2(u^8 - u^4 + 1)/3 + u, \\ r = u^8 - u^4 + 1, \\ t = u + 1. \end{cases}$$

The final exponentiation for BLS24 curves is decomposed into two parts thanks to the cyclotomic polynomial

$$\frac{q^{24} - 1}{r} = \left(q^{12} - 1\right)\left(q^4 + 1\right)\frac{q^8 - q^4 + 1}{r}.$$

The hard part of the final exponentiation can be decomposed in basis q [73] as:

$$\frac{q^8 - q^4 + 1}{r} = \sum_{i=0}^{\phi(24)-1} \Lambda_i q^i = \Lambda_0 + \Lambda_1 q + \Lambda_2 q^2 + \cdots + \Lambda_7 q^7,$$

where

$$\begin{cases} \Lambda_0 = u^9 - 2u^8 + u^7 - u^5 + 2u^4 - u^3 + 3, \\ \Lambda_1 = u^8 - 2u^7 + u^6 - u^4 + 2u^3 - u^2, \\ \Lambda_2 = u^7 - 2u^6 + u^5 - u^3 + 2u^2 - u, \\ \Lambda_3 = u^6 - 2u^5 + u^4 - u^2 + 2u - 1, \\ \Lambda_4 = u^5 - 2u^4 + u^3, \\ \Lambda_5 = u^4 - 2u^3 + u^2, \\ \Lambda_6 = u^3 - 2u^2 + u, \\ \Lambda_7 = u^2 - 2u + 1. \end{cases}$$

The best result in the literature to our knowledge is the one presented in [36]. In their work, the hard part of the final exponentiation is presented as follows:

$$\begin{aligned} \Lambda_0 &= \Lambda_1 u + 3, & \Lambda_1 &= \Lambda_2 u, \\ \Lambda_2 &= \Lambda_3 u, & \Lambda_3 &= \Lambda_4 u - \Lambda_7, \\ \Lambda_4 &= \Lambda_5 u, & \Lambda_5 &= \Lambda_6 u, \\ \Lambda_6 &= \Lambda_7 u, & \Lambda_7 &= u^2 - 2u + 1. \end{aligned}$$

The overall cost of the hard part of the final exponentiation is then 8 exponentiations by u, one exponentiation by $u/2$ (since u is even), one squaring, 10 multiplications and 7-Frobenius operations in $\mathbb{F}_{q^{24}}$. Then, we need to add two Frobenius operations, two multiplications and one inversion in $\mathbb{F}_{q^{24}}$ to compute the final exponentiation. For computing the Optimal ate pairing over BLS24 curves for the 128 bit security level, we consider the arithmetic presented in [3] and the parameter $u = -2^{32} + 2^{28} + 2^{12}$ proposed in Section 4 the final exponentiation requires 18732 multiplications and 10 Inversions in $\mathbb{F}_q$ when considering the compressed squaring and 23400 multiplications and one inversion when the cyclotomic squaring is considered.
For computing the Optimal ate pairing over BLS24 curves for the 192 bit security level, we consider the parameter $u = -2^{56} - 2^{43} + 2^9 - 2^6$ proposed in Section 4.9 the final exponentiation requires 27985 multiplications and 10 Inversions in $\mathbb{F}_q$ when considering the compressed squaring and 36573 multiplications and one inversion when the cyclotomic squaring is considered.
For computing the Optimal ate pairing over BLS24 curves for the 256 bit security level, we consider the parameter $u = 2^{103}2^{101} + 2^{68} + 2^{50}$ proposed in Section 4.9 the final exponentiation requires 43213 multiplications and 10 Inversions in $\mathbb{F}_q$ when considering the compressed squaring and 59415 multiplications and one inversion when the cyclotomic squaring is considered.

5.4 The case of $k = 27$

Elliptic curves of embedding degree $k = 27$ are suitable for computing Miller loop. In this paragraph, we give the computation of the final exponentiation on this category of curves which is defined by the parameter u as follow [81]

$$\begin{cases} q = 1/3(u-1)^2(u^{18} + u^9 + 1) + u, \\ r = 1/3(u^{18} + u^9 + 1), \\ t = u + 1. \end{cases}$$

In this case, the final exponentiation consists on computing

$$\frac{q^{27} - 1}{r} = (q^9 - 1)\frac{q^{18} + q^9 + 1}{r}.$$

Then, the representation of the hard part of the final exponentiation can be given as described in [81] as follow.

$$(u-1)^2 \times (q^9 + u^9 + 1) \times (q^8 + uq^7 + u^2q^6 + u^3q^5 + \cdots + u^7q + u^8) + 3.$$

This decomposition requires one inversion in $\mathbb{F}_{q^{27}}$, 17 exponentiations by u, 2 exponentiations by $(u-1)$, 11 multiplications, 2 q^9, q, q^2, q^3, q^4, q^5, q^6, q^7 and q^8 Frobenius maps. When

considering our parameter $u = 2^3 + 3^4 + 2^{11} + 2^{15}$ given in Section 4 the overall cost of the final exponentiation for computing the final exponentiation for the 128-bit security level is then 76980 multiplications and one inversion in $\mathbb{F}_q$.
For the 192-bit security level, we consider the parameter $u = 2^{22} + 2^{14} + 2^9 + 2^8 + 2^4 + 2^3 + 2$ proposed in Section 4.9, and then, the cost of computing the final exponentiation of the Optimal ate pairing is about 100730 multiplications and one inversion in $\mathbb{F}_q$.

For the 256 bit security level, we consider the parameter $u = 2^{29} + 2^{17} + 2^{15} + 2^{11} + 2^4 + 1$ proposed in Section 4.9, and then, the cost of computing the final exponentiation of the Optimal ate pairing is about 115000 multiplications and one inversion in $\mathbb{F}_q$.
In the following Tables, we summarize the cost of the final exponentiation of the Optimal Ate pairing in the target elliptic curves for each security level: 128, 192 and 256.

6 Conclusion and recommendations

In this article we update the key size for pairing-based cryptography according to the latest discrete logarithm attack. We unify the results according of the NFS attack and apply them to more than 150 pairing-friendly elliptic curves. Our motivation was that the NFS attack is more efficient on BN and BLS 12 elliptic curves which were the most popular for the implementation of pairing due to their efficient arithmetic. Once we obtain the security evaluation of the curves, we compare the efficiency of the optimal Ate computation on them. To do so, we first give an estimation for the Miller loop, and we evaluate the final exponentiation for the most promising curves. Indeed, the Miller loop alone is not sufficient to evaluate the complexity of the pairing computation as the final exponentiation represents the half of optimal Ate pairing computation. We evaluate the final exponentiation only for curves with a very efficient Miller loop, the criteria of efficiency being the complexity of the Miller loop for the BLS-12.
Table 17 presents the cost of the optimal Ate pairing for our short list of candidates at the 128, 192, and 256-bits security level.

Some informal remarks

We deliberately avoided to use our insight to eliminate bad candidates because we wanted to be sure that we don't miss any good pairing. We can however make a list of *a posteori* informal remarks:

- At 128 bits of security, among the good candidates in Table 17, the bit size of the target field varies between 5281 and 7642 bits, which represents a 45% difference. A larger field means a larger cost of the arithmetic, but this remains less than the factor three which is the advantage of multiples of 6 when compared to degrees which are coprime to 6. All the good pairings at 128 bits of security in Table 17 are multiples of 6.
- Fifteen is not the new twelve.[5] A simplified manner to choose k is to take $k\rho$ equal to the bit size of the target field, which is now about 5000 bits for 128 bits of security, divided by the lower bound on r which is 256. Hence one could have set $k\rho = 20$ and, for many BLS pairings $\rho = 1.33$ so a possible guess of k is 15. But the above remark says that 12 and 24 are better candidates because they are multiples of 6.
- $k = 27$ at 256 bits is a compromise between good arithmetic and strength against the TNFS attack.[6] Indeed, 27 has a unique divisor between 2 and 8 so an NFS attack can be done in a restricted number of manners. At 256 bits of security, an ideal situation would be to have 5, 6 or 7 as a divisor, so 3 is a bad approximation of the optimal parameters. Hence BLS-27

[5] In a personal communication Tanja Lange asked the first author if 15 is the new 12.
[6] As a direct remark on the exTNFS attack, Pierrick Gaudry told the first author that a good candidate would be a compromise: a degree k which is not coprime to 6 but which has few small divisors, e.g. $k = 2p$ with $p \geq 5$ prime.

resists well to the TNFS attack albeit not as good as a pairing of prime degree. In the same time 27 is not coprime to 6 so it has a fast pairing.
– BLS-24 is the new challenger of BLS-12 at the 128 bits security level. The detailed analysis of the security, the complexity of the Miller loop and the final exponentiation shows that the two pairings are relatively similar. A detail which attracts our attention is the cost of the arithmetic in a field $\mathbb{F}_{q^{12}}$ vs. $\mathbb{F}_{q^{24}}$ when the field has approximately 100 machine words. The arithmetic in $\mathbb{F}_q$ is done using a schoolbook algorithm because q has few machine words, whereas the field extension arithmetic is done using Karatsuba tricks. This could help BLS-24 to be a good alternative to BLS-12.

To conclude, if one wants an efficient pairing implementation using existing arithmetic that will support several security levels the BLS 24 curve is the one to be chosen, even if at the 256 security level BLS 27 is twice more efficient according to our estimation. On another hand, if we are willing to find the most efficient pairing, further works are necessary to improve the final exponentiation for the BLS 27 family. It is possible that with a more efficient final exponentiation, the BLS 27 would provide the most efficient pairing at the 192 or 128-bits security level. Indeed, the Miller algorithm for BLS 27 is very efficient at each security level, the overall cost of pairing is penalized by the final exponentiation.

References

1. Zcash. https://www.zfnd.org/.
2. 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*, volume 7708 of *Lecture Notes in Computer Science*, 2012.
3. 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.
4. Giuseppe Ateniese, Kevin Fu, Matthew Green, and Susan Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Trans. Inf. Syst. Secur.*, 9(1), February 2006.
5. Razvan Barbulescu and Sylvain Duquesne. Updating key size estimations for pairings. *J. of Cryptology. Published online at https://link.springer.com/article/10.1007%2Fs00145-018-9280-5*, 2018.
6. Razvan Barbulescu, Pierrick Gaudry, Aurore Guillevic, and François Morain. Improving NFS for the discrete logarithm problem in non-prime finite fields. In *Advances in Cryptology - Eurocrypt 2015*, volume 9056 of *Lecture Notes in Computer Science*, 2015.
7. Razvan Barbulescu, Pierrick Gaudry, and Thorsten Kleinjung. The Towed Number Field Sieve. In *Advances in Cryptology – Asiacrypt 2015*, volume 9453 of *Lecture Notes in Computer Science*, 2015.
8. Razvan Barbulescu, Nadia El Mrabet, and Loubna Ghammam. A taxonomy of pairings, their security, their complexity. Cryptology ePrint Archive, Report 2018/485, 2018.
9. Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Constructing elliptic curves with prescribed embedding degrees. In *Security in Communication Networks*, 2002.
10. Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In *Selected Areas in Cryptography - SAC 2005*, 2005.
11. Jean-Luc Beuchat, Jorge E. González-Díaz, Shigeo Mitsunari, Eiji Okamoto, Francisco Rodríguez-Henríquez, and Tadanori Teruya. High-speed software implementation of the optimal Ate pairing over Barreto–Naehrig curves. In *Pairing-Based Cryptography – Pairing 2010*, volume 6487 of *Lecture Notes in Computer Science*, 2010.
12. Jean-Luc Beuchat, Emmanuel López-Trejo, Luis Martínez-Ramos, Shigeo Mitsunari, and Francisco Rodríguez-Henríquez. Multi-core implementation of the Tate pairing over supersingular elliptic curves. In *Cryptology and Network Security – CANS 2009*, volume 5888 of *Lecture Notes in Computer Science*, 2009.

13. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, 2004.
14. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *LNCS*, 2001.
15. Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Advances in Cryptology - Eurocrypt 2003*, volume 2656 of *Lecture Notes in Computer Science*, 2003.
16. Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *Advances in Cryptology - CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, 2005.
17. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. *J. of Cryptology*, 17(4), 2004.
18. Sébastien Canard, Aïda Diop, Nizar Kheir, Marie Paindavoine, and Mohamed Sabt. BlindIDS: Market-compliant and privacy-friendly intrusion detection system over encrypted traffic. In *Asia Conference on Computer and Communications Security*. ACM, 2017.
19. Steve Chang. Trend micro. http://www.trendmicro.fr, 2008.
20. Craig Costello, Tanja Lange, and Michael Naehrig. Faster pairing computations on curves with high-degree twists. In *Public Key Cryptography – PKC 2010*, volume 6056 of *Lecture Notes in Computer Science*, 2010.
21. Quentin Deschamps, Aurore Guillevic, and Shashank Singh. Estimating size requirements for pairings: Simulating the tower-NFS algorithm in GF(p^n), 2017. Slides are available at https://ecc2017.cs.ru.nl/slides/ecc2017-guillevic.pdf.
22. Augusto Jun Devegili, Michael Scott, and Ricardo Dahab. Implementing cryptographic pairings over Barreto-Naehrig curves. In *Pairing-Based Cryptography – Pairing 2007*, volume 4575 of *Lecture Notes in Computer Science*, 2007.
23. Robert Dryło. On constructing families of pairing-friendly elliptic curves with variable discriminant. In *Progress in cryptology – INDOCRYPT 2011*, volume 7107 of *Lecture Notes in Computer Science*, 2011.
24. Pu Duan, Shi Cui, and Choong Wah Chan. Special polynomial families for generating more suitable elliptic curves for pairing-based cryptosystems. In *The 5th WSEAS International Conference on Electronics, Hardware, Wireless and Optimal Communications*, 2005.
25. Sylvain Duquesne, Nadia El Mrabet, Safia Haloui, and Franck Rondepierre. Choosing and generating parameters for low level pairing implementation on BN curves. *Appl. Algebra Eng. Commun. Comput.*, 29(2), 2018.
26. Sylvain Duquesne and Loubna Ghammam. Memory-saving computation of the pairing final exponentiation on BN curves. *Groups Complexity Cryptology*, 8(1), 2016.
27. Nadia El Mrabet, Aurore Guillevic, and Sorina Ionica. Efficient multiplication in finite field extensions of degree 5. In *Progress in Cryptology - AFRICACRYPT 2011 - 4th International Conference on Cryptology in Africa, Dakar, Senegal, July 5-7, 2011. Proceedings*, pages 188–205, 2011.
28. H. Fan and M. A. Hasan. Comments on "five, six, and seven-term karatsuba-like formulae. *IEEE Transactions on Computers*, 56(5):716–717, 2007.
29. Georgios Fotiadis and Elisavet Konstantinou. Generating pairing-friendly elliptic curve parameters using sparse families. *Journal of Mathematical Cryptology*, 12(2), 2018.
30. 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.
31. Georgios Fotiadis and Chloe Martindale. Optimal TNFS-secure pairings on elliptic curves with composite embedding degree. Cryptology ePrint Archive, Report 2019/555, 2019.
32. Emmanuel Fouotsa, Nadia El Mrabet, and Aminatou Pecha. Computing optimal ate pairings on elliptic curves with embedding degree 9, 15 and 27. *IACR Cryptology ePrint Archive*, 2016.
33. Emmanuel Fouotsa, Nadia El Mrabet, and Aminatou Pecha Njiahouo. Computing optimal ate pairings on elliptic curves with embedding degree 9,15 and 27. Cryptology ePrint Archive, Report 2016/1187, 2016.

34. David Freeman, Michael Scott, and Edlyn Teske. A taxonomy of pairing-friendly elliptic curves. *J. of Cryptology*, 23(2), 2010.
35. Laura Fuentes Castañeda, Edward Knapp, and Francisco Rodríguez Henríquez. Faster hashing to $\mathbb{G}_2$. In *Selected Areas in Cryptography – SAC 2011*, volume 2259 of *Lecture Notes in Computer Science*, 2011.
36. Loubna Ghammam and Emmanuel Fouotsa. Improving the computation of the optimal ate pairing for a high security level. *J. Appl. Math. Comput.*, 1, 2018.
37. Laurent Grémy. Higher-dimensional sieving for the number field sieve algorithms. *The Open Book Series*, 2(1):275–291, 2019.
38. Gurleen Grewal, Reza Azarderakhsh, Patrick Longa, Shi Hu, and David Jao. Efficient implementation of bilinear pairings on ARM processors. In *Selected Areas in Cryptography – SAC 2012*, volume 7707 of *Lecture Notes in Computer Science*, 2012.
39. Aurore Guillevic. A short-list of stnfs-secure pairing-friendly curves at the 128-bit security level. *IACR Cryptology ePrint Archive*, 2019:1371, 2019.
40. Aurore Guillevic. A short-list of pairing-friendly curves resistant to Special TNFS at the 128-bit security level. In *Public key cryptology – PKC 2020*, 2020.
41. Aurore Guillevic, Simon Masson, and Emmanuel Thomé. Cocks–Pinch curves of embedding degrees five to eight and optimal ate pairing computation. *Designs, Codes and Cryptography*, pages 1–35, 2020.
42. Aurore Guillevic and Shashank Singh. A comparison of pairing-friendly curves at the 192-bit security level, 2019. Available online at https://members.loria.fr/AGuillevic/files/talks/19_Roscoff_STNFS.pdf.
43. J. Jeong and T. Kim. Extended tower number field sieve with application to finite fields of arbitrary composite extension degree. Cryptology ePrint Archive, Report 2016/526, 2016. http://eprint.iacr.org/2016/526.
44. Antoine Joux. A one round protocol for tripartite Diffie-Hellman. In *Algorithmic Number Theory (ANTS-IV)*, volume 1838 of *Lecture Notes in Computer Science*, 2000.
45. Antoine Joux, Reynald Lercier, Nigel P. Smart, and Frederik Vercauteren. The number field sieve in the medium prime case. In *Advances in Cryptology - CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, 2006.
46. Antoine Joux and Cécile Pierrot. The special number field sieve in $\mathbb{F}_{p^n}$ – application to pairing-friendly constructions. In *Pairing-Based Cryptography - Pairing 2013*, volume 8365 of *Lecture Notes in Computer Science*, 2013.
47. Marc Joye and Gregory Neven, editors. *Identity-Based Cryptography*, volume 2 of *Cryptology and Information Security Series*. IOS press, 2009.
48. Ezekiel J. Kachisa, Edward F. Schaefer, and Michael Scott. Constructing Brezing-Weng pairing-friendly elliptic curves using elements in the cyclotomic field. In *Pairing-Based Cryptography – Pairing 2008*, 2008.
49. Md. Al-Amin Khandaker, Yuki Nanjo, Loubna Ghammam, Sylvain Duquesne, Yasuyuki Nogami, and Yuta Kodera. Efficient optimal ate pairing at 128-bit security level. In *Progress in Cryptology - INDOCRYPT 2017*, volume 10698 of *Lecture Notes in Computer Science*, 2017.
50. T. Kim and R. Barbulescu. The extended tower number field sieve: A new complexity for the medium prime case. In *Advances in Cryptology – CRYPTO 2016*, volume 9814 of *Lecture notes in computer science*, 2016.
51. T KLEINJUNG. Cofactorisation strategies for the number field sieve and an estimate for the sieving step for factoring 1024-bit integers. In European Network of Excellence (ECRYPT), editor, *Special-purpose Hardware for Attacking Cryptographic Systems Workshop– SHARCS 2006*, 2006.
52. Thorsten Kleinjung. On polynomial selection for the general number field sieve. *Math. Comp.*, 75(256), 2006.
53. Thorsten Kleinjung. Polynomial selection, 2008. In CADO workshop on integer factorization, INRIA Nancy, 2008. http://cado.gforge.inria.fr/workshop/slides/kleinjung. pdf.

54. Donald E. Knuth. *The Art of Computer Programming, Volume 1 (3rd Ed.): Fundamental Algorithms*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1997.
55. Duc-Phong Le, Nadia El Mrabet, Safia Haloui, and Chik How Tan. On the near prime-order mnt curves. *Applicable Algebra in Engineering, Communication and Computing*, 30(2), 2019.
56. Arjen K Lenstra. Unbelievable security: Matching AES security using public key systems. In *Advances in cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, 2001.
57. Xibin Lin, Chang-An Zhao, Fangguo Zhang, and Yanming Wang. Computing the ate pairing on elliptic curves with embedding degree k= 9. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 91(9), 2008.
58. Ben Lynn. Mnt curves. https://crypto.stanford.edu/pbc/mnt.html.
59. Dmitrii Viktorovich Matyukhin. Effective version of the number field sieve for discrete logarithm in a field $GF(p^k)$. *Trudy po Diskretnoi Matematike*, 9, 2006.
60. Gary McGuire and Oisin Robinson. A new angle on lattice sieving for the number field sieve. arXiv preprint arXiv:2001.10860, 2020.
61. Alfred Menezes, Palash Sarkar, and Shashank Singh. Challenges with assessing the impact of NFS advances on the security of pairing-based cryptography. In *Paradigms in Cryptology - Mycrypt 2016*, volume 10311 of *Lecture Notes in Computer Science*, 2016.
62. Atsuko Miyaji, Masaki Nakabayashi, and Shunzo Takano. Characterization of elliptic curve traces under FR-reduction. In *ICISC*, volume 2015 of *Lecture Notes in Computer Science*, 2000.
63. Peter L. Montgomery. Five, six, and seven-term karatsuba-like formulae. *IEEE Trans. Computers*, 54(3):362–369, 2005.
64. Dustin Moody and Lily Chen. Pairing project. https://csrc.nist.gov/Projects/Pairing-Based-Cryptography, 2011.
65. D. Page, N. P. Smart, and F. Vercauteren. A comparison of MNT curves and supersingular curves. *Applicable Algebra in Engineering, Communication and Computing*, 17(5), 2006.
66. John M Pollard. Factoring with cubic integers. In *The development of the number field sieve*, volume 1554 of *Lecture Notes in Mathematics*, pages 4–10. Springer, 1993.
67. Francisco Rodríguez-Henríquez and Erkay Savas. Special issue in honor of peter lawrence montgomery. *J. Cryptographic Engineering*, 8(3):185–187, 2018.
68. P. Sarkar and S. Singh. Fine tuning the function field sieve algorithm for the medium prime case. *IEEE Transactions on Information Theory*, 62(4), 2016.
69. P. Sarkar and S. Singh. A generalisation of the conjugation method for polynomial selection for the extended tower number field sieve algorithm. Cryptology ePrint Archive, Report 2016/537, 2016.
70. Palash Sarkar and Shashank Singh. New complexity trade-offs for the (multiple) number field sieve algorithm in non-prime fields. In *Advances in Cryptology – Eurocrypt 2016*, volume 9665 of *Lecture Notes in Computer Science*, 2016.
71. Michael Scott. Missing a trick: Karatsuba variations. *Cryptography and Communications*, 10(1):5–15, 2018.
72. Michael Scott and Paulo S. L. M. Barreto. Generating more mnt elliptic curves. *Designs, Codes and Cryptography*, 38(2):209–217, Feb 2006.
73. Michael Scott, Naomi Benger, Manuel Charlemagne, Luis J. Dominguez Perez, and Ezekiel J. Kachisa. On the final exponentiation for calculating pairings on ordinary elliptic curves. In *Pairing-Based Cryptography – Pairing 2009*, 2009.
74. Michael Scott and Aurore Guillevic. A new family of pairing-friendly elliptic curves. In *Finite Fields arithmetic – WAIFI 2018*, Lecture Notes in Computer Science, 2018.
75. Mike Scott. Missing a trick: Karatsuba revisited. *IACR Cryptology ePrint Archive*, 2015:1247, 2015.
76. Caroline Sheedy. *Privacy Enhancing Protocols using Pairing Based Cryptography*. PhD thesis, Dublin City University, 2010.

77. Justine Sherry, Chang Lan, Raluca Ada Popa, and Sylvia Ratnasamy. Blindbox: Deep packet inspection over encrypted traffic. *ACM SIGCOMM Computer communication review*, 45(4), 2015.
78. Executive Team. Voltage security. https://www.voltage.com, 2005.
79. T. Unterluggauer and E. Wenger. Practical attack on bilinear pairings to disclose the secrets of embedded devices. In *2014 Ninth International Conference on Availability, Reliability and Security*, Sept 2014.
80. Meng Zhang and Maozhi Xu. Generating pairing-friendly elliptic curves using parameterized families. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 101(1), 2018.
81. Xusheng Zhang and Dongdai Lin. Analysis of optimum pairing products at high security levels. In *Progress in Cryptology – INDOCRYPT 2012*, volume 7668 of *Lecture Notes in Computer Science*, 2012.

Method 6.2						
k	min $\log_2 q$	min $\log_2 u$	u	$\log_2 q$	Miller's cost	$\approx$
9	484	22	$-1+2^3+2^4+2^5+2^9+2^{10}+2^{22}$	482	$44\mathbf{D}+20\mathbf{A}+1\mathbf{D}+M_k+I_k$	31 155$M_q + I_k$
11	336	13	$-1+2^8+2^{14}$	363	$28\mathbf{D}+4M\mathbf{A}+1\mathbf{D}+M_k+I_k$	65 316$M_q + I_k$
13	351	12	$1+2+2^3+2^4+2^8+2^{10}+2^{14}+2^{20}$	599	$20\mathbf{D}+14\mathbf{A}+1\mathbf{D}+M_k+I_k$	110 085$M_q + I_k$
15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35				complexity higher than 203 985$M_q + I_k$		
37, 39, 43, 45			no value for u below 2^{11}			
Method 6.3						
10	572	40	$1+2^3+2^4+2^8+2^{39}+2^{40}$	432	$79\mathbf{D}+15M\mathbf{A}+\mathbf{L}+M_k$	23 816
14	382	21	$1-2^2+2^6+2^9-2^{12}-2^{15}-2^{19}+2^{22}$	390	$44\mathbf{D}+14M\mathbf{A}+\mathbf{L}+M_k$	12 228
18	467	21	$1+2+2^3+2^5+2^7+2^8+2^{10}+2^{12}+2^{13}+2^{22}$	482	$44\mathbf{D}+11M\mathbf{A}+\mathbf{L}+M_k$	23 458
22	497	19	$1+2+2^4+2^{14}+2^{15}$	403	$30\mathbf{D}+9M\mathbf{A}+\mathbf{L}+M_k$	78 423
26	321	11	$1+2^8+2^{12}$	360	$24\mathbf{D}+5M\mathbf{A}+\mathbf{L}+M_k$	81 248
30	542	16	$1+2^2+2^3-2^{10}+2^{14}+2^{16}$	552	$32\mathbf{D}+11M\mathbf{A}+\mathbf{L}+M_k$	26 687
34	307	8	$1-2^4+2^{10}+2^{14}$	533	$28\mathbf{D}+6M\mathbf{A}+\mathbf{L}+M_k$	165 138
38	356	5	$1+2^3+2^9+2^{11}+2^{17}$	713	$34\mathbf{D}+11M\mathbf{A}+\mathbf{L}+M_k$	268 200
42	508	11	$1+2^4+2^7+2^8+2^{10}+2^{11}$	539	$24\mathbf{D}+7M\mathbf{A}+\mathbf{L}+M_k$	225 150
46	302	6	$1+2+2^9+2^{10}+2^{13}$	660	$26\mathbf{D}+9M\mathbf{A}+\mathbf{L}+M_k$	315 415
50	453	9	$1+2^4-2^7+2^{10}+2^{11}+2^{14}$	746	$28\mathbf{D}+9M\mathbf{A}+\mathbf{L}+M_k$	50 603
54	395	7	$1+2+2^3+2^5+2^8+2^9+2^{11}$	664	$23\mathbf{D}+9M\mathbf{A}+\mathbf{L}+M_k$	74 466
Method 6.4						
12	510	63, 7	$1+2+2^3+2^8+2^9+2^{11}+2^{64}$	510	$64\mathbf{D}+6M\mathbf{A}+\mathbf{L}+M_k$	10 141
20	382	31, 8	$1+2^4+2^{16}+2^{32}$	383	$32\mathbf{D}+3M\mathbf{A}+\mathbf{L}+M_k$	9 116
28	350	21, 8	$1+2+2^3+2^4+2^8+2^9+2^{22}$	350	$22\mathbf{D}+6M\mathbf{A}+\mathbf{L}+M_k$	10 278
36	438	21, 9	$1+2^2+2^{10}+2^{14}+2^{16}+2^{22}$	438	$22\mathbf{D}+5M\mathbf{A}+\mathbf{L}+M_k$	18 901
44	310	12, 9	$1+2^7+2^8+2^{12}+2^{14}$	342	$14\mathbf{D}+4M\mathbf{A}+\mathbf{L}+M_k$	59480
52	306	10, 9	$1-2^6+2^9+2^{12}+2^{13}$	380	$13\mathbf{D}+4M\mathbf{A}+\mathbf{L}+M_k$	81134
Method 6.7, degree 2 twist						
12	445	32	$1+2^{14}+2^{17}+2^{32}$	445	$64\mathbf{D}+9M\mathbf{A}+\mathbf{L}+M_k$	13 976
24	381	4.4	$1+2^2+2^8+2^9+2^{32}$	381	$32\mathbf{D}+4M\mathbf{A}+\mathbf{L}+M_k$	20 192
30	550	10	$1+2+2^5-2^7+2^{12}$	691	$48\mathbf{D}+25M\mathbf{A}+\mathbf{L}+M_k$	56 133
36	541	16	$1+2^3+2^5-2^8+2^{11}+2^{13}+2^{16}$	547	$32\mathbf{D}+13M\mathbf{A}+\mathbf{L}+M_k$	56 963
42	667	9	no value for u below 2^{11}			
48	525	24	$1+2^3+2^5+2^6+2^8+2^{10}+2^{14}+2^{24}$	525	$24\mathbf{D}+7M\mathbf{A}+\mathbf{L}+M_k$	72 348
Method 6.7, without twists						
12	445	32	$1+2^{14}+2^{17}+2^{32}$	445	$64\mathbf{D}+9M\mathbf{A}+\mathbf{L}+M_k$	13 976
24	381	4.4	$1+2^2+2^8+2^9+2^{32}$	381	$32\mathbf{D}+4M\mathbf{A}+\mathbf{L}+M_k$	20 192
30	550	10	$1+2+2^5-2^7+2^{12}$	691	$48\mathbf{D}+25M\mathbf{A}+\mathbf{L}+M_k$	56 133
36	541	16	$1+2^3+2^5-2^8+2^{11}+2^{13}+2^{16}$	547	$32\mathbf{D}+13M\mathbf{A}+\mathbf{L}+M_k$	56 963
42	667	9	no value for u below 2^{11}			
48	525	24	$1+2^3+2^5+2^6+2^8+2^{10}+2^{14}+2^{24}$	525	$24\mathbf{D}+7M\mathbf{A}+\mathbf{L}+M_k$	72 348
9	507	11	$-1+2^4+2^5+2^9+2^{11}$	520	$48\mathbf{D}+15M\mathbf{A}+M\mathbf{A}+M_k+I_k$	31 369$M_q + I_k$
15	607	9	$1+2^2+2^4+2^7+2^9+2^{10}+2^{12}+2^{14}$	950	$56\mathbf{D}+34M\mathbf{A}+6M\mathbf{A}+10M_k+I_k$	85 050$M_q + I_k$
21	598	7	$1+2+2^4+2^7-2^{10}+2^{13}$	1100	$52\mathbf{D}+23M\mathbf{A}+M\mathbf{A}+M_k+I_k$	97 135$M_q + I_k$
27	465	4	$1-2^2-2^7+2^{10}+2^{11}$	1218	$48\mathbf{D}+16M\mathbf{A}+M\mathbf{A}+M_k+I_k$	157 460$M_q + I_k$
33, 39, 45			no value for u below 2^{10}			

Table 10: Methods 6.2, 6.3, 6.4 and 6.7 at 128 bits of security

Degree 6 twist						
k	min $\log_2 q$	min $\log_2 u$	u	$\log_2 q$	Miller's cost	$\approx M_q$
12	461	64	$-2^{77}+2^{50}+2^{33}$	460	$77\mathbf{D}+2M\mathbf{A}+\mathbf{L}+M_k$	7 438
24	318	32	$-2^{32}+2^{28}+2^{12}$	319	$32\mathbf{D}+2M\mathbf{A}+\mathbf{L}+M_k$	9 381
30	383	32	$2^{32}+2^{14}+2^{13}+2^3+1$	383	$32\mathbf{D}+4M\mathbf{A}+\mathbf{L}+M_k$	9 887
42	350	22	$-2^{22}+2^{18}+2^6$	349	$22\mathbf{D}+2M\mathbf{A}+\mathbf{L}+M_k$	9 738
48	286	16	$2^{16}+2^{14}+2^{13}+2^{11}+2^6$	296	$17\mathbf{D}+4M\mathbf{A}+\mathbf{L}+M_k$	17 042
Degree 2 twist						
k	min $\log_2 q$	min $\log_2 u$	u	$\log_2 q$	Miller's cost	$\approx$
14	350	21, 9	$-1+2^6+2^7+2^9+2^{10}+2^{13}+2^{17}+2^{22}$	352	$44\mathbf{D}+11M\mathbf{A}+2\mathbf{L}+3\pi_q+3M_k$	$11\ 173M_q$
16	350, 5	16	$2^3+2^5+2^6-2^8+2^{11}-2^{14}+2^{17}$	369	$66\mathbf{D}+4M\mathbf{A}+2\mathbf{L}+3\pi_q+3M_k+I_k$	$28\ 282M_q+I_k$
20	350, 65	16	$1+2^6+2^{17}$	372	$34\mathbf{D}+4M\mathbf{A}+2\mathbf{L}+3\pi_q+3M_k$	$15\ 990M_q$
22	364	13	2^5+2^{17}	474	$34\mathbf{D}+2M\mathbf{A}+2\mathbf{L}+3\pi_q+3M_k+I_k$	$64\ 426M_q+I_k$
26	306, 6	10, 9	$2^2+2^3+2^5+2^7+2^{13}+2^{14}$	407	$28\mathbf{D}+8M\mathbf{A}+2\mathbf{L}+3\pi_q+3M_k$	$91\ 242M_q$
28	373	10, 9	$-2^2+2^7+2^8+2^{10}+2^{14}$	478	$28\mathbf{D}+8M\mathbf{A}+2\mathbf{L}+3\pi_q+3M_k+I_k$	$21\ 778M_q+I_k$
32	280	8, 3	$2+2^4+2^5+2^9$	309	$20\mathbf{D}+6M\mathbf{A}+2\mathbf{L}+3\pi_q+3M_k$	$32\ 990M_q$
34	354	8, 8	$2+2^3+2^5+2^{10}$	400	$20\mathbf{D}+3M\mathbf{A}+2\mathbf{L}+3\pi_q+3M_k+I_k$	$102\ 102M_q+I_k$
38	356	8, 9	$1+2^2+2^4+2^6+2^7+2^{10}$	409	$20\mathbf{D}+9M\mathbf{A}+2\mathbf{L}+3\pi_q+3M_k$	$152\ 518M_q$
40	370	8	$-2^3+2^7+2^{10}$	466	$20\mathbf{D}+3M\mathbf{A}+2\mathbf{L}+3\pi_q+3M_k+I_k$	$28\ 984M_q+I_k$
44, 46, 50, 52			no value for u below 2^{12}			
Degree 3 twist						
k	min $\log_2 q$	min $\log_2 u$	u	$\log_2 q$	Miller's cost	$\approx$
15	382, 4	31, 8	$1+2^2+2^{12}+2^{16}+2^{32}$	383	$32\mathbf{D}+4M\mathbf{A}+\mathbf{L}+M_k$	$11\ 173M_q$
21	350, 4	21, 9	$2^4+2^6+2^9+2^{12}+2^{15}+2^{22}$	351	$44\mathbf{D}+11M\mathbf{A}+$ extra computation	$19160M_q+I_k$
27	298, 5	15, 1	$2^3+2^4+2^{11}+2^{15}$	300	$15\mathbf{D}+3M\mathbf{A}$	$6\ 401M_q$
33	311	13	$1+2+2^7+2^9+2^{14}$	336	$14\mathbf{D}+4M\mathbf{A}+\mathbf{L}+M_k$	$54\ 320M_q$
39	308	11	$2^4+2^7+2^{10}+2^{11}+2^{13}$	375	$26\mathbf{D}+9M\mathbf{A}+$ extra computation	$145\ 000M_q+I_k$
45	351	11	$1+2-2^3+2^8+2^{10}+2^{11}$	373	$12\mathbf{D}+8M\mathbf{A}+M\mathbf{A}+M_k$	$17\ 832M_q$
Without twist						
k	min $\log_2 q$	min $\log_2 u$	u	$\log_2 q$	Miller's cost	$\approx$
11	311	13	$2^4+2^6+2^7+2^9+2^{10}+2^{14}$	338	$27\mathbf{D}+10\mathbf{A}+I_k+M_k$	$84\ 538M_q+I_k$
13	308	11	$2^4+2^7+2^{10}+2^{11}+2^{13}$	376	$26\mathbf{D}+6\mathbf{A}+I_k+M_k$	$125\ 722M_q+I_k$
29	643	10, 7	$2^4-2^7+2^{10}+2^{11}$	690	$22\mathbf{D}+7\mathbf{A}+I_k+M_k$	$511\ 589M_q+I_k$
17, 19, 23, 25, 31, 35, 37, 41, 43, 47, 49, 51, 53			no value for u below 2^{12}			

Table 11: Method 6.6 (BLS), 128 bits of security, twist of degree 6, 3, 2 and 1 (no twist)

k	$\log_{32}(q)$	Miller's in m_{32}	$\log_{64}(q)$	Miller's in m_{64}
12	15	1 673 550	8	476 032
24	10	938 100	5	234 525

Table 12: Method 6.6, Comparison of the best candidates

Operation	Complexity in projective coordinates
Doubling step [81]	$M_{3b} + kM_q + 3M_e + 9S_e + M_k + S_k$
Mixed addition [81]	$kM_q + 12M_e + 5S_e + M_k$
Final line evaluation	$(5k-4)M_q + S_q + S_{k/d} + M_{k/d} + 2M\mathbf{A}$

Table 13: Complexity of Miller's steps using twist of degree 3

KSS							
k	min $\log_2 q$	min $\log_2 u$	u	$\log_2 q$	$\approx$ Miller's cost	m_{32}	m_{64}
16	330	33	$-2^{34} + 2^{27} - 2^{23} + 2^{20} - 2^{11} + 1$	340	$7\ 534M_q$	911 614	271 224
18	356	44	$2^{44} + 2^{22} - 2^9 + 2$	352	$9\ 431M_q$	1 358 064	339516
32	344	19	$2^5 + 2^{10} + 2^{11} + 2^{19} + 2^{20}$	349	$19\ 321M_q + I_k$	2 337841	695 556
36	321	23	$1 + 2 + 2^4 + 2^9 + 2^{14} + 2^{17} + 2^{23} + 2^{24}$	329	$10\ 771M_q + I_k$	1 303 291	387 756
40	376	17	$1 + 2^4 + 2^7 + 2^8 + 2^{13} + 2^{18}$	377	$18\ 254M_q$	2 628 576	657 144
54	315, 9	15, 7	$2^3 + 2^7 + 2^{11} + 2^{15} + 2^{16}$	348	$20\ 427M_q$	2 471 667	735 372
other families							
k	min $\log_2 q$	min $\log_2 u$	u	$\log_2 q$	$\approx$ Miller' cost	m_{32}	m_{64}
9	590	73	$2^{74} + 2^{35} - 2^{22} + 3$	590	$8\ 808M_q$	2 050 048	512 512
$BN-12$	472	118	$-1 - 2^{14} + 2^{101} + 2^{114}$	456	12 068	2 715 300	772 352
15	383	31, 9	$2 + 2^{10} + 2^{16} + 2^{19} + 2^{32}$	383	$6\ 836M_q$	984 384	246 096

Table 14: KSS, MNT, BN and other curves, 128 bits of security

128 bits							
Method	k	u	$\log_2 q$	$\log_2(q^k)$	$\approx M_q$	m_{32}	m_{64}
6.3	14	$1-2^2+2^6+2^9-2^{12}-2^{15}-2^{19}+2^{22}$	390	5 460	12 228	2 066 532	599 172
6.4	20	$1+2^4+2^{16}+2^{32}$	383	7 660	9 116	1 312 704	328 176
6.4	28	$1+2+2^3+2^4+2^8+2^9+2^{22}$	350	9 800	10 278	1 243 638	370 008
6.6	12	$-2^{77}+2^{50}+2^{33}$	461	5 520	7 438	1 673 550	476 032
6.6	15	$1+2^2+2^{12}+2^{16}+2^{32}$	383	5 745	11 173	1 608 912	402 228
6.6	24	$-2^{32}+2^{28}+2^{12}$	319	7 656	9 381	938 100	234 525
6.6	27	$2^3+2^4+2^{11}+2^{15}$	300	8 058	6 401	640 100	160 025
6.7	12	$1+2^{14}+2^{17}+2^{32}$	445	5 340	13 976	2 739 296	684 824
KSS	16	$-2^{34}+2^{27}-2^{23}+2^{20}-2^{11}+1$	340	5 540	7 534	911 614	271 224
DCC	15	$2+2^{10}+2^{16}+2^{19}+2^{32}$	383	5 745	6 836	984 384	246 096

192 bits							
Method	k	u	$\log_2 q$	$\log_2(q^k)$	$\approx M_q$	m_{32}	m_{64}
6.3	14	$1-2^3+2^7+2^8+2^{11}+2^{40}$	719	10 053	21 940	11 606 260	3 159 360
6.4	20	$1+2^6+2^9+2^{11}+2^{12}+2^{16}+2^{61}$	731	14 601	16 735	8 852 815	2 409 840
6.4	28	$-2^{31}-2^{13}-2^1-1$	495	13 833	13 250	3 392 000	848 000
6.6	12	$2+2^2+2^6+2^7+2^{11}+2^{105}+2^{206}+2^{207}$	1 244	14 928	28 831	43 851 951	11 532 400
6.6	15	$1-2^8+2^{12}+2^{15}+2^{16}-2^{72}+2^{75}$	897	13 442	13 320	11 202 120	2 997 000
6.6(ZL)	27	$2^{22}+2^{14}+2^9+2^8+2^4+2^3+2$	438, 5	11 841	16 178	2 734 082	792 722
6.6	24	$-2^{56}-2^{43}+2^9-2^6$	559	13 403	16 368	4 730 352	1 325 808
6.7	24	$-2^{48}+2^{12}+2^{42}+1$	573	13 746	38 871	12 594 204	3 148 551
KSS	16	$2^4-2^6+2^{12}+2^{13}+2^{15}+2^{16}+2^{25}+2^{81}+2^{83}$	834	13 332	24 795	6 347 520	1 586 880
KSS	18	$2-2^5+2^9+2^{11}+2^{14}+2^{82}$	657	11 809	13 488	5 948 208	1 632 048
DCC	15	$2^3+2^8+2^{16}-2^{18}+2^{21}+2^{77}$	927	13 891	13 507	12 156 300	3 039 075

256 bits, exotic k							
Method	k	u	$(\log_2(q))$	$\log_2 q^k$	$\approx M_q$	m_{32}	m_{64}
6.2	17	$-1-2^6+2^{11}+2^{13}+2^{14}+2^{32}$	1215	20 639	40 277	58×10^6	14×10^6
6.3	18	$-1+2^6+2^8+2^{10}+2^{12}+2^{14}+2^{43}$	945	16 993	43 479	39 131 100	9 782 775
	19	$-2^4+-2^{10}+2^{13}+2^{15}$	610	11 587	55 455	22 182 000	5 545 500
	20	$2-2^4-2^9+2^{14}+2^{46}$	1011	8 914	20 209	48 958 464	12 239 616
6.6	22	$1-2^2+2^6+2^7+2^{11}+2^{29}$	813	17 865	39 479	26 687 804	6 671 951
(BLS)	26	$2^2+2^8+2^{12}+2^{17}+2^{23}$	644	16 720	38 857	15 542 800	3 885 700
	28	$2+2^2+2^4+2^6+2^8-2^{11}+2^{17}+2^{22}$	748	20 942	60 380	34 778 880	8 694 720
6.7	9	$1+2^3-2^5-2^{10}+2^{13}+2^{14}+2^{20}+2^{21}$	990		61 373	58 979 453	15 711 488

256 bits							
Method	k	u	$(\log_2(q))$	$\log_2 q^k$	$\approx M_q$	m_{32}	m_{64}
6.4	20	$1+2^3-2^6+2^{10}-2^{12}+2^{15}+2^{77}+2^{78}+2^{79}$	956	19 114	21 723	19 550 700	4 887 675
6.4	28	$1+2+2^7+2^8-2^{10}+2^{15}+2^{62}$	991	27 721	25 314	24 326 754	6 480 384
6.6	15	$2^{10}+2^{11}+2^{30}+2^{150}$	1799	26 977	32 736	102 669 504	25 667 376
6.6	24	$2^{103}-2^{101}+2^{68}+2^{50}$	1026	24 621	37 126	38 017 024	9 504 256
6.6	27	$-2^8+2^{12}+2^{21}+2^{29}$	579	15 621	18 493	5 991 732	1 497 933
6.7	12	$-1+2^4+2^9-2^{12}+2^{15}+2^{119}$	1664	19 957	57 279	$> 154.10^6$	$> 38.10^6$
KSS	16	$1+2^6-2^{10}-2^{12}-2^{13}+2^{16}+2^{18}+2^{149}$	1733	27 719	38 904	85 938 936	22 408 704
KSS	18	$2-2^3-2^7-2^{12}+2^{15}+2^{16}+2^{20}+2^{174}$	1391	25 028	28 026	82 698 333	21 563 712
LZZW	9	$2^4+2^7-2^{12}-2^{17}+2^{19}+2^{287}$	2295	20 650	35 755	185 353 920	46 338 480
DCC	15	$1+2+2^3+2^5+2^6-2^8+2^{15}+2^{112}$	1345	20161	22 435	39 575 340	10 775 835

Table 15: Miller loop : comparison of the best candidates for 128, 192 and 256 bits of security

128 bits						
Method	k	u	$(\log_2(q))$	$\approx M_q$	m_{32}	m_{64}
6.4	20	$1+2^4+2^{16}+2^{32}$	383	$29\ 250+I$	4 212 144	1 053 036
	28	$1+2+2^3+2^4+2^8+2^9+2^{22}$	350	$50\ 302+I$	6 086 663	1 810 908
6.6	12	$-2^{77}+2^{50}+2^{33}$	460	$6\ 188+6I$	1 393 650	396 416
	15	$1+2+2^{12}+2^{16}+2^{32}$	383	$19\ 738+I$	2 842 416	710 604
	24	$-2^{32}+2^{28}+2^{12}$	319	$18\ 732+10I$	1 874 200	468 550
	27	$2^3+2^4+2^{11}+2^{15}$	300	$76\ 980+I$	7 698 100	1 924 525
KSS	16	$-2^{34}+2^{27}-2^{23}+2^{20}-2^{11}+1$	340	$18\ 514+I$	2 240 315	666 540
DCC	15	$2+2^{10}+2^{16}+2^{19}+2^{32}$	383	$19\ 190+I$	2 763 504	690 876
BN	12	$2^{114}+2^{101}-2^{14}-1$	462	$5\ 598+4I$	1 260 450	358 528
192 bits						
Method	k	u	$(\log_2(q))$	$\approx M_q$	m_{32}	m_{64}
6.4	20	$1+2^6+2^9+2^{11}+2^{12}+2^{16}+2^{61}$	733	$57\ 762+I$	30 556 627	8 317 872
	28	$-2^{31}-2^{13}-2^1-1$	494	$121\ 550+I$	31 117 056	7 779 264
6.6	12	$2+2^2+2^6+2^7+2^{11}+2^{105}+2^{206}+2^{207}$	1 246	$15\ 500+6I$	23 584 626	6 202 400
	15	$1-2^8+2^{12}+2^{15}+2^{16}-2^{72}+2^{75}$	899	$42\ 707+I$	35 917 428	9 609 300
$6.6(ZL)$	27	$2^{22}+2^{14}+2^9+2^8+2^4+2^3+2$	$438,5$	$100\ 730+I$	19 743 276	4 100 731
6.6	24	$-2^{56}-2^{43}+2^9-2^6$	518	$27\ 985+10I$	6 662 810	2 267 595
KSS	16	$2^4-2^6+2^{12}+2^{13}+2^{15}+2^{16}+2^{25}+2^{81}+2^{83}$	500	$36\ 000+I$	9 216 256	2 304 064
	18	$2-2^5+2^9+2^{11}+2^{14}+2^{82}$	652	$24\ 719+8I$	10 904 607	2 991 967
DCC	15	$2^3+2^8+2^{16}-2^{18}+2^{21}+2^{77}$	929	$41\ 942+I$	37 748 700	9 437 175
256 bits						
Method	k	u	$(\log_2(q))$	$\approx M_q$	m_{32}	m_{64}
6.4	20	$1+2^3-2^6+2^{10}-2^{12}+2^{15}+2^{77}+2^{78}+2^{79}$	958	$75\ 582+I$	68 024 700	17 006 175
	28	$1+2+2^7+2^8-2^{10}+2^{15}+2^{62}$	989	$175\ 550+I$	168 704 511	44 641 056
6.6	15	$2^{10}+2^{11}+2^{30}+2^{150}$	1780	$79\ 337+I$	248 803 968	62 200 992
	24	$2^{103}-2^{101}+2^{68}+2^{50}$	1024	$43\ 213+10I$	44 260 352	11 065 088
	27	$1+2^4+2^{11}+2^{15}+2^{17}+2^{29}$	568	$115\ 000+I$	37 260 324	9 315 081
KSS	16	$1+2^6-2^{10}-2^{12}-2^{13}+2^{16}+2^{18}+2^{149}$	1485	$43\ 128+I$	95 271 961	24 842 304
	18	$2-2^3-2^7-2^{12}+2^{15}+2^{16}+2^{20}+2^{174}$	1392	$41\ 687+8I$	80 721 520	20 180 380
DCC	15	$1+2+2^3+2^5+2^6-2^8+2^{15}+2^{112}$	1342	$60\ 257+I$	106 295 112	26 573 775

Table 16: Final exponentiation : Comparison of the best candidates for 128, 192 and 256 bits of security

128 bits							
Method	k	u	Miller	Final Expo.	$\approx M_q$	m_{32}	m_{64}
6.3	14	$-1-2^4+2^7-2^{11}+2^{15}+2^{22}$	12 228	$17\ 702 + I$	$29\ 931 + I$	5 058 508	1 466 668
6.4	20	$1+2^4+2^{16}+2^{32}$	9 116	$29\ 250 + I$	$38\ 366 + I$	5 524 848	1 381 212
6.6	12	$-2^{77}+2^{50}+2^{33}$	7438	$6\ 188 + 6I$	$13\ 626 + 6I$	3 067 200	872 448
	15	$1+2+2^{12}+2^{16}+2^{32}$	11 173	$19\ 738 + I$	$30\ 911 + I$	4 025 520	1 306 387
	24	$-2^{32}+2^{28}+2^{12}$	9 381	$18\ 732 + 10I$	$28\ 113 + 10I$	2 812 300	703 075
	27	$2^3+2^4+2^{11}+2^{15}$	6 401	$76\ 980 + I$	$83\ 381 + I$	8 338 200	2 084 550
KSS	16	$-2^{34}+2^{27}-2^{23}+2^{20}-2^{11}+1$	7 534	$18\ 514 + I$	$26\ 048 + I$	3 151 929	937 764
DCC	15	$2+2^{10}+2^{16}+2^{19}+2^{32}$	6 836	$19\ 190 + I$	$26\ 026 + I$	3 747 885	936 972
BN	12	$2^{114}+2^{101}-2^{14}-1$	12 068	$5\ 598 + 4I$	$17600 + 4I$	3 961 800	1 126 860
192 bits							
Method	k	u	Miller	Final Expo.	$\approx M_q$	m_{32}	m_{64}
6.4	20	$1 + 2^6 + 2^9 + 2^{11} + 2^{12} + 2^{16} + 2^{61}$	16 735	$57\ 762 + I$	$72\ 497 + I$	38 351 442	9 587 872
	28	$-2^{31} - 2^{13} - 2^1 - 1$	13 250	$121\ 550 + I$	$134\ 800 + I$	34 508 800	8 627 200
6.6	12	$2+2^2+2^6+2^7+2^{11}+2^{105}+2^{206}+2^{207}$	28 831	$15\ 500 + 6I$	$44\ 331 + 6I$	67 436 577	16 859 145
	15	$1 - 2^8 + 2^{12} + 2^{15} + 2^{16} - 2^{72} + 2^{75}$	13 320	$42\ 707 + I$	$56\ 027 + I$	47 119 548	11 779 887
	24	$-2^{56} - 2^{43} + 2^9 - 2^6$	16 368	$27\ 985 + 10I$	$44\ 353 + 10I$	12 820 907	3 205 227
6.6[81]	27	$2^{22} + 2^{14} + 2^9 + 2^8 + 2^4 + 2^3 + 2$	16 178	$100\ 730 + I$	$116\ 908 + I$	22 928 276	5 732 069
KSS	16	$2^4-2^6+2^{12}+2^{13}+2^{15}+2^{16}+2^{25}+2^{81}+2^{83}$	24 795	$36\ 000 + I$	$60\ 795 + I$	15 564 032	3 891 008
	18	$2 - 2^5 + 2^9 + 2^{11} + 2^{14} + 2^{82}$	13 488	$24\ 719 + 8I$	$38\ 207 + 8I$	16 852 815	4 213 204
DCC	15	$2^3 + 2^8 + 2^{16} - 2^{18} + 2^{21} + 2^{77}$	13 507	$41\ 942 + I$	$55\ 449 + I$	49 905 900	12 476 475
256 bits							
Method	k	u	Miller	Final Expo.	$\approx M_q$	m_{32}	m_{64}
6.4	20	$1+2^3-2^6+2^{10}-2^{12}+2^{15}+2^{77}+2^{78}+2^{79}$	21 723	$75\ 582 + I$	$97\ 305 + I$	87 575 400	21 893 850
	28	$1+2+2^7+2^8-2^{10}+2^{15}+2^{62}$	25 314	$175\ 550 + I$	$200\ 864 + I$	193 031 265	48 257 816
6.6	15	$2^{10} + 2^{11} + 2^{30} + 2^{150}$	32 736	$79\ 337 + I$	$112\ 073 + I$	351 464 064	87 866 016
	24	$2^{103}-2^{101}+2^{68}+2^{50}$	37 126	$43\ 213 + 10I$	$80\ 339 + 8I$	82 276 352	20 569 088
	27	$-2^8 + 2^{12} + 2^{21} + 2^{29}$	18 493	$115\ 000 + I$	$134\ 647 + I$	43 625 952	10 906 488
KSS	16	$1+2^6-2^{10}-2^{12}-2^{13}+2^{16}+2^{18}+2^{149}$	38 904	$43\ 128 + I$	$82\ 032 + I$	144 706 212	36 176 553
	18	$2-2^3-2^7-2^{12}+2^{15}+2^{16}+2^{20}+2^{174}$	28 026	$41\ 687 + 8I$	$69\ 713 + 8I$	134 987 600	33 746 900
DCC	15	$1+2+2^3+2^5+2^6-2^8+2^{15}+2^{112}$	22 435	$60\ 257 + I$	$82\ 692 + I$	145 870 452	36 467 613

Table 17: Overall cost of the optimal Ate pairing