Simulating the TNFS algorithm to deduce cryptographic key-sizes for field extensions $GF(p^n)$

Aurore Guillevic

Université de Lorraine, CNRS, Inria, LORIA, Nancy, France
aurore.guillevic@inria.fr

Workshop, November 24, 2023

https://members.loria.fr/AGuillevic/files/talks/23_FMorain.pdf

Discrete logarithm problem

$\mathbb{G}$ multiplicative group of order ℓ
g generator, $\mathbb{G} = \{1, g, g^2, g^3, \ldots, g^{\ell-2}, g^{\ell-1}\}$

Given $h \in \mathbb{G}$, find integer $x \in \{0, 1, \ldots, \ell - 1\}$ such that $h = g^x$.
Exponentiation easy: $(g, x) \mapsto g^x$
Discrete logarithm hard in well-chosen groups $\mathbb{G}$
Common choices of $\mathbb{G}$:

- prime finite field $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$ (1976)
- characteristic 2 field $\mathbb{F}_{2^n}$ ($\approx$ 1979)
- elliptic curve $E(\mathbb{F}_p)$ (1985)

Choosing key sizes

Symmetric ciphers (AES): key sizes are 128, 192 or 256 bits.
Perfect symmetric cipher: trying all keys of size n bits takes 2^n tests
$\rightarrow$ **brute-force search**

perfect symmetric cipher with secret key in $[0, 2^n - 1]$, of n bits $\leftrightarrow$ n bits of security

For DL-based key exchange with p, ℓ of length(p), length(ℓ) bits:
n bits of security $\leftrightarrow$ the best (mathematical) attack should take at least 2^n steps

- what is the fastest attack?
- how much time does it take with respect to length(p), length(ℓ)?

RSA and Diffie–Hellman keys are much larger.
Cipher suite: a pair of symmetric and asymmetric ciphers offering the same level of security.

Discrete log problem

How fast can we invert the exponentiation function $(g, x) \mapsto g^x$?

- $g \in G$ generator, $\exists$ always a preimage $x \in \{1, \ldots, \#G\}$
- naive search, try them all: $\#G$ tests
- $O(\sqrt{\#G})$ generic algorithms
- independent search in each distinct subgroup + CRT (Pohlig-Hellman)

Discrete log problem

How fast can we invert the exponentiation function $(g, x) \mapsto g^x$?

- $g \in G$ generator, $\exists$ always a preimage $x \in \{1, \ldots, \#G\}$
- naive search, try them all: $\#G$ tests
- $O(\sqrt{\#G})$ generic algorithms
- independent search in each distinct subgroup + CRT (Pohlig-Hellman)

$\rightarrow$ choose G of large prime order (no subgroup)

$\rightarrow$ complexity of inverting exponentiation in $O(\sqrt{\#G})$

$\rightarrow$ security level 128 bits means $\sqrt{\#G} \geq 2^{128}$
take $\#G = 2^{256}$
analogy with symmetric crypto, keylength 128 bits (16 bytes)

Discrete log problem

How fast can we invert the exponentiation function $(g, x) \mapsto g^x$?

- $g \in G$ generator, $\exists$ always a preimage $x \in \{1, \ldots, \#G\}$
- naive search, try them all: $\#G$ tests
- $O(\sqrt{\#G})$ generic algorithms
- independent search in each distinct subgroup + CRT (Pohlig-Hellman)

$\rightarrow$ choose G of large prime order (no subgroup)

$\rightarrow$ complexity of inverting exponentiation in $O(\sqrt{\#G})$

$\rightarrow$ security level 128 bits means $\sqrt{\#G} \geq 2^{128}$
take $\#G = 2^{256}$
analogy with symmetric crypto, keylength 128 bits (16 bytes)

Use additional structure of G if any.

Discrete log problem when $\mathbb{G} = (\mathbb{Z}/p\mathbb{Z})^*$

Index calculus algorithm [Western–Miller 68, Adleman 79], prequel of the Number Field Sieve algorithm (NFS)

- p prime, $(p-1)/2$ prime, $\mathbb{G} = (\mathbb{Z}/p\mathbb{Z})^*$, gen. g, target h
- get many multiplicative relations in $\mathbb{G}$
- get one multiplicative relation involving the target h
- take logarithms: linear relations *in the exponents*
- solve a linear system to get discrete logarithms
- get $x = \log h$

Index calculus in $(\mathbb{Z}/p\mathbb{Z})^*$

Multiplicative relations over the **integers**

Smooth integers $n = p_1^{e_1} p_2^{e_2} \cdots p_i^{e_i}$, $p_i \leq B$ are quite common $\rightarrow$ it works
Complexity $e^{\sqrt{(2+o(1))(\log p)(\log\log p)}}$ (Pomerance 87)

Index calculus in $(\mathbb{Z}/p\mathbb{Z})^*$

Multiplicative relations over the **integers**

Smooth integers $n = p_1^{e_1} p_2^{e_2} \cdots p_i^{e_i}$, $p_i \leq B$ are quite common $\rightarrow$ it works
Complexity $e^{\sqrt{(2+o(1))(\log p)(\log \log p)}}$ (Pomerance 87)

Improvements in the 80's, 90's:

- Sieve (faster relation collection)
- Smaller integers to factor
- Multiplicative relations in **number fields**
- Better **sparse linear algebra**
- Independent targets h

Number Field

1985: ElGamal, DL in $\mathrm{GF}(p^2)$ with two quadratic number fields
1986: Coppersmith–Odlyzko–Schroeppel, DL algorithm in $\mathrm{GF}(p)$
1995: Weber–Denny, record computation 85 dd with $\mathbb{Q}[\sqrt{-2}]$

Number Field

1985: ElGamal, DL in GF(p^2) with two quadratic number fields
1986: Coppersmith–Odlyzko–Schroeppel, DL algorithm in GF(p)
1995: Weber–Denny, record computation 85 dd with $\mathbb{Q}[\sqrt{-2}]$

- If $p = 1 \bmod 4$, exists u, v s.t. $p = u^2 + v^2$, $\theta = \sqrt{-1}$
- If $p = 3 \bmod 8$, exists u, v s.t. $p = u^2 + 2v^2$, $\theta = \sqrt{-2}$
- If $p = 7 \bmod 8$, exists u, v s.t. $p = u^2 - 2v^2$, $\theta = \sqrt{2}$

and $|u|, |v| < \sqrt{p}$
$u/v \equiv m \mod p$ and $m^2 + s = 0 \mod p$

Number Field

1985: ElGamal, DL in GF(p^2) with two quadratic number fields
1986: Coppersmith–Odlyzko–Schroeppel, DL algorithm in GF(p)
1995: Weber–Denny, record computation 85 dd with $\mathbb{Q}[\sqrt{-2}]$

- If $p = 1 \bmod 4$, exists u, v s.t. $p = u^2 + v^2$, $\theta = \sqrt{-1}$
- If $p = 3 \bmod 8$, exists u, v s.t. $p = u^2 + 2v^2$, $\theta = \sqrt{-2}$
- If $p = 7 \bmod 8$, exists u, v s.t. $p = u^2 - 2v^2$, $\theta = \sqrt{2}$

and $|u|, |v| < \sqrt{p}$
$u/v \equiv m \mod p$ and $m^2 + s = 0 \mod p$

Define a map from $\mathbb{Z}[\theta]$ to $\mathbb{Z}/p\mathbb{Z}$

$$\begin{array}{rcl} \phi\colon \mathbb{Z}[\theta] & \to & \mathbb{Z}/p\mathbb{Z} \\ \theta & \mapsto & m \bmod p \text{ where } m = u/v,\ m^2 + s = 0 \bmod p \end{array}$$

ring homomorphism $\phi(a + b\theta) = a + bm$

$$\phi(\underbrace{a + b\theta}_{\substack{\text{factor in} \\ \mathbb{Z}[\theta]}}) = a + bm = (a + b\ \underbrace{u/v}_{=m}) = (\underbrace{av + bu}_{\text{factor in } \mathbb{Z}})v^{-1} \mod p$$

Commutative diagram for NFS

$$\begin{tikzcd}
 & \mathbb{Z}[x] \arrow[dl, "x \mapsto m"'] \arrow[dr, "x \mapsto \alpha"] & \\
a - bm \in \quad \mathbb{Z} \arrow[dr] & & \mathbb{Z}[\alpha] \text{ subring of } \mathbb{Q}(\alpha) \ni a - b\alpha \arrow[dl, "\alpha \mapsto m"] \\
 & \mathbb{Z}/p\mathbb{Z} &
\end{tikzcd}$$

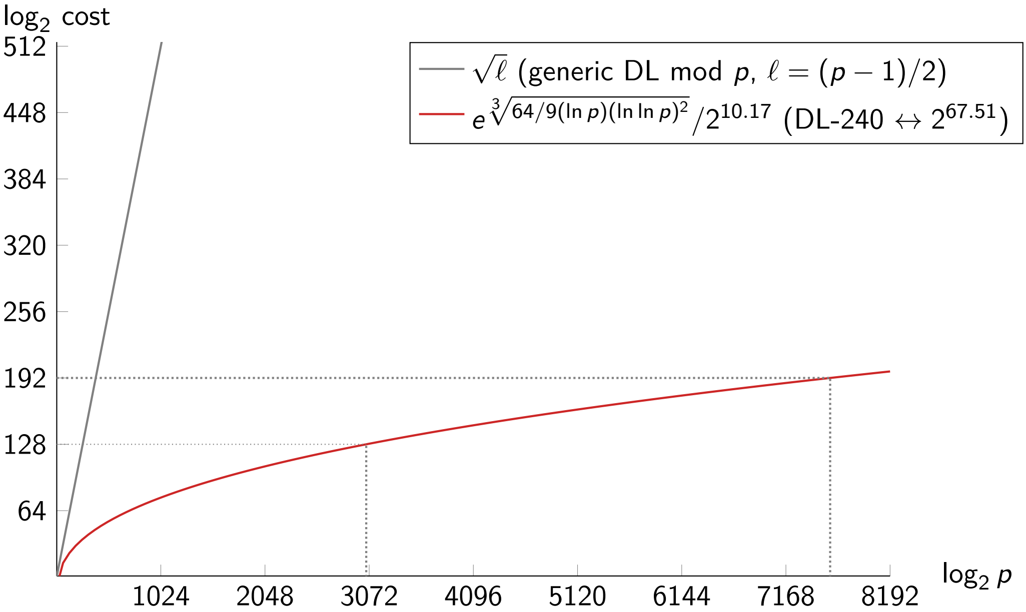
Number Field Sieve

Since 1993 (Gordon, Schirokauer):

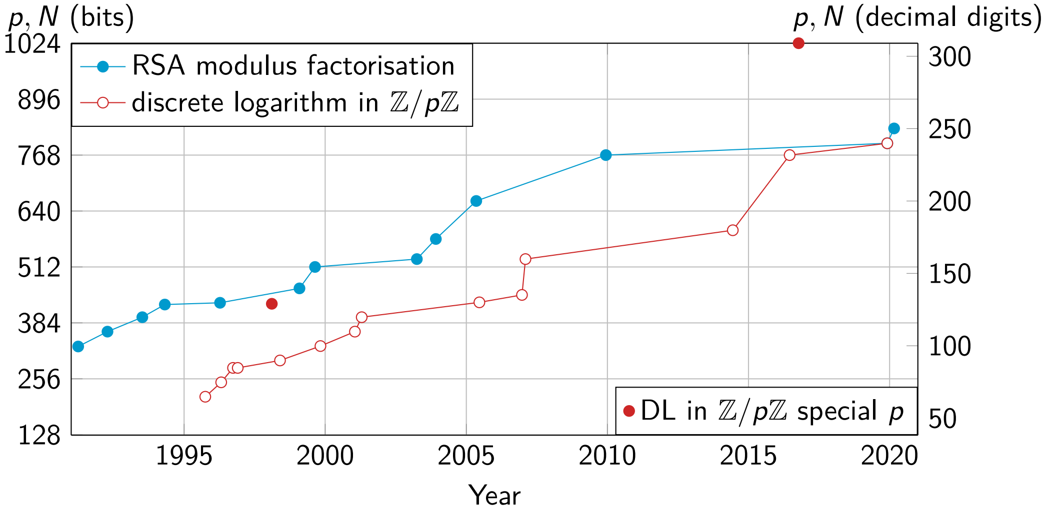
$$L_p(1/3, c) = \exp\left((c + o(1))(\log p)^{1/3}(\log\log p)^{2/3}\right)$$

- polynomial selection
- **relation collection** $L_p(1/3, 1.923)$
  sieve to enumerate efficiently (a, b) pairs
- **sparse linear algebra** $L_p(1/3, 1.923)$
  compute right kernel mod prime ℓ, block-Wiedemann alg.
- individual discrete logarithm

Latest record computation:
240 decimal digits (dd) i.e. 795-bit prime $p = \text{RSA-240} + 49204$, $\ell = (p-1)/2$ prime
Boudot, Gaudry, G., Heninger, Thomé, Zimmermann, 2019 [BGG$^+$20]
Total time: 3177 core-years on Intel Xeon Gold 6130 2.1GHz


log2 cost
512
448
384
320
256
192
128
64
$\sqrt{\ell}$ (generic DL mod p, $\ell = (p-1)/2$)
$e^{\sqrt[3]{64/9(\ln p)(\ln\ln p)^2}}/2^{10.17}$ (DL-240 $\leftrightarrow$ $2^{67.51}$)
1024
2048
3072
4096
5120
6144
7168
8192
log2 p

Record computations



Discrete Log in $\mathbb{F}_{p^k}$

$\mathbb{F}_{p^k}$ much less investigated than $\mathbb{F}_p$ or integer factorization
Much better results in pairing-related fields

- Special NFS in $\mathbb{F}_{p^k}$: Joux–Pierrot 2013 [JP14]
- Tower NFS (TNFS): Barbulescu–Gaudry–Kleinjung 2015 [BGK15]
- Extended Tower NFS: Kim–Barbulescu [KB16], Kim–Jeong [KJ17], Sarkar–Singh 2016 [SS16]

Use more structure: subfields

Special Tower NFS

$\mathbb{F}_{p^{2k}}$, subfield $\mathbb{F}_{p^2}$ defined by y^2+1
Idea: $a+bx$ in NFS $\rightarrow$ $(a_0+a_1 i)+(b_0+b_1 i)x$ in TNFS
Integers to factor are **much smaller**

- factors integer $\mathrm{Norm}_f = \mathrm{Res}(\mathrm{Res}(\mathbf{a}+\mathbf{b}x, f_y(x)), y^2+1)$
- factors integer $\mathrm{Norm}_g = \mathrm{Res}(\mathrm{Res}(\mathbf{a}+\mathbf{b}x, g_y(x)), y^2+1)$

Res = resultant of polynomials

Special Tower NFS

$\mathbb{F}_{p^{2k}}$, subfield $\mathbb{F}_{p^2}$ defined by y^2+1
Idea: $a+bx$ in NFS $\to$ $(a_0+a_1 i)+(b_0+b_1 i)x$ in TNFS
Integers to factor are **much smaller**

- factors integer $\mathrm{Norm}_f = \mathrm{Res}(\mathrm{Res}(\mathbf{a}+\mathbf{b}x, f_y(x)), y^2+1)$
- factors integer $\mathrm{Norm}_g = \mathrm{Res}(\mathrm{Res}(\mathbf{a}+\mathbf{b}x, g_y(x)), y^2+1)$

Res = resultant of polynomials
$p = p(s)$ is special

Special Tower NFS

$\mathbb{F}_{p^{2k}}$, subfield $\mathbb{F}_{p^2}$ defined by y^2+1
Idea: $a+bx$ in NFS $\rightarrow$ $(a_0+a_1 i)+(b_0+b_1 i)x$ in TNFS
Integers to factor are **much smaller**

- factors integer $\mathrm{Norm}_f = \mathrm{Res}(\mathrm{Res}(\mathbf{a}+\mathbf{b}x, f_y(x)), y^2+1)$
- factors integer $\mathrm{Norm}_g = \mathrm{Res}(\mathrm{Res}(\mathbf{a}+\mathbf{b}x, g_y(x)), y^2+1)$

Res = resultant of polynomials
$p = p(s)$ is special

Index calculus in the 80's: implemented *before* complexity known
TNFS: complexity known, implementation just started for $\mathrm{GF}(p^6), \mathrm{GF}(p^4)$

- DL in $\mathrm{GF}(p^6)$ of 521 bits with TNFS, De Micheli, Gaudry, Pierrot [DGP21]
- DL in $\mathrm{GF}(p^4)$ of 512 bits with TNFS, Robinson, 2022

Variants of NFS: Complexities

large characteristic $p = L_{p^n}(\alpha),\ \alpha > 2/3$:	
$(64/9)^{1/3} \simeq 1.923$	NFS
special p:	
$(32/9)^{1/3} \simeq 1.526$	SNFS

medium characteristic $p = L_{p^n}(\alpha),\ 1/3 < \alpha < 2/3$:	
$(96/9)^{1/3} \simeq 2.201$	prime n NFS-HD (Conjugation [BGGM15])
$(48/9)^{1/3} \simeq 1.747$	composite n (Kim–Barbulescu 2016 [KB16]), best case of TNFS: when parameters fit perfectly
special p:	
$(64/9)^{1/3} \simeq 1.923$	NFS-HD+Joux–Pierrot'13 [JP14]
$(32/9)^{1/3} \simeq 1.526$	composite n, best case of STNFS (Kim–Barbulescu 2016 [KB16])

Special Tower NFS

1. Polynomial selection: choose 3 polynomials h, f, g
2. Relation collection: obtain many smooth norms of
   $\boldsymbol{a} + \boldsymbol{b}\theta_f = (a_0 + a_1\tau + \ldots + a_i\tau^i) + (b_0 + b_1\tau + \ldots + b_i\tau^i)\theta_f$, $\boldsymbol{a} + \boldsymbol{b}\theta_g$
3. Filtering step of the matrix (apply Galois automorphisms if any)
4. Linear algebra
5. Individual discrete logarithm

Are the norms as smooth as integers of the same size?
Bias $\rightarrow \alpha(f), \alpha(g)$
TNFS: $\alpha(h, f), \alpha(h, g)$

Simulation without sieving

Polynomial selection: for many pairs (f, g)

- compute $\alpha(h, f), \alpha(h, g)$ (w.r.t. subfield) **bias in smoothness**
- select polys f, g with negative bias $\alpha(f), \alpha(g)$ if possible
- **Monte-Carlo** simulation with 10^6 random samples from
  $\mathcal{S} = \{(a_0 + a_1 y + \ldots + a_d y^d) + (b_0 + b_1 y + \ldots + b_d y^d)x,\ |a_i|, |b_j| < A\}$
  For each sample:
  1. compute its algebraic norm N_f, N_g in each number field
  2. smoothness probability (N_f, α_f), (N_g, α_g) with Dickman-ρ
- Average smoothness probability of samples
  $\rightarrow$ estimation of the total number of possible relations in $\mathcal{S}$
  $\rightarrow$ **Murphy's E for TNFS**

Simulation without sieving

Polynomial selection: for many pairs (f, g)

- compute $\alpha(h, f), \alpha(h, g)$ (w.r.t. subfield) **bias in smoothness**
- select polys f, g with negative bias $\alpha(f), \alpha(g)$ if possible
- **Monte-Carlo** simulation with 10^6 random samples from
  $\mathcal{S} = \{(a_0 + a_1 y + \ldots + a_d y^d) + (b_0 + b_1 y + \ldots + b_d y^d)x,\ |a_i|, |b_j| < A\}$
  For each sample:
  1. compute its algebraic norm N_f, N_g in each number field
  2. smoothness probability (N_f, α_f), (N_g, α_g) with Dickman-ρ
- Average smoothness probability of samples
  $\rightarrow$ estimation of the total number of possible relations in $\mathcal{S}$
  $\rightarrow$ **Murphy's E for TNFS**

dichotomy to approach the best balanced parameters

smoothness bound B, coefficient bound A.

$\rightarrow$ refinement of Barbulescu–Duquesne technique [BD19]

Example : Barreto-Naehrig curve, p 254 bits

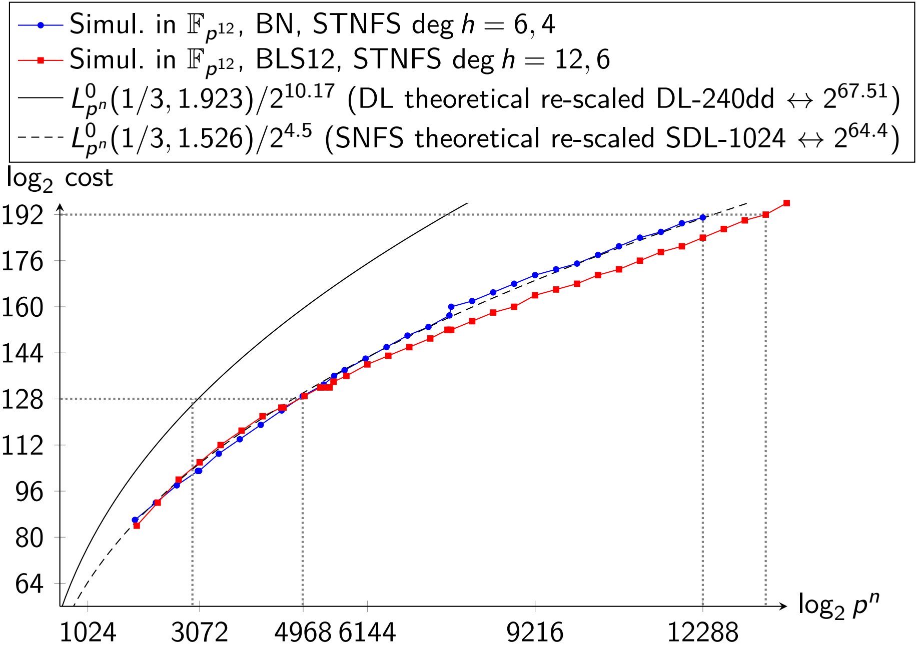
$p = 36s^4 + 36s^3 + 24s^2 + 6s + 1$ where $s = -(2^{62} + 2^{55} + 1)$

$f = 36x^8 + 36yx^6 + 24y^2x^4 + 6y^3x^2 + y^4$

$g = x^2 + sy = x^2 + 4647714815446351873y$

$B = 2000$

h	$1/\zeta_{K_h}(2)$	$\alpha(h, f, B)$	$\alpha(h, g, B)$	$\alpha_f + \alpha_g$
$y^6 + y^5 - y^2 - y - 1$	0.953	2.042	2.479	4.521
$y^6 - y^4 + y^3 + y^2 - 1$	0.917	1.288	1.740	3.028
$y^6 + y^3 + y^2 - y - 1$	0.917	2.419	2.876	5.295
$y^6 + y^5 - y^3 + y - 1$	0.909	0.278	2.357	2.636
$y^6 + y^5 + y^4 + y^3 + y^2 + y - 1$	0.883	2.341	2.033	4.374
$y^6 + y^4 + y^3 + y - 1$	0.867	0.899	2.526	3.425
$y^6 + y^4 + y^2 + y + 1$	0.836	1.955	1.141	3.095
$y^6 + y^5 + y^2 - y + 1$	0.763	0.891	1.264	2.155
$y^6 + y^5 - y^4 + y^3 + y^2 + y - 1$	0.756	0.956	1.177	2.133
$y^6 + y^5 + y - 1$	0.736	1.925	2.108	4.032
$y^6 + y^5 + y^3 - y^2 + y - 1$	0.732	1.729	2.099	3.828
$y^6 + y^3 + y - 1$	0.728	-0.250	1.191	0.941
$y^6 + y^3 - y + 1$	0.720	1.605	1.348	2.952
$y^6 + y^3 + y^2 + 1$	0.718	1.151	1.294	2.445
$y^6 - y^4 + y^3 - y^2 - y - 1$	0.710	0.406	2.278	2.684
$y^6 + y^5 - y^3 + y^2 - y + 1$	0.697	1.572	0.818	2.390
$y^6 + y^4 + y + 1$	0.679	1.319	1.683	3.002



Simul. in $\mathbb{F}_{p^{12}}$, BN, STNFS deg $h = 6, 4$
Simul. in $\mathbb{F}_{p^{12}}$, BLS12, STNFS deg $h = 12, 6$
$L^0_{p^n}(1/3, 1.923)/2^{10.17}$ (DL theoretical re-scaled DL-240dd $\leftrightarrow 2^{67.51}$)
$L^0_{p^n}(1/3, 1.526)/2^{4.5}$ (SNFS theoretical re-scaled SDL-1024 $\leftrightarrow 2^{64.4}$)
$\log_2$ cost
192
176
160
144
128
112
96
80
64
1024
3072
4968
6144
9216
12288
$\log_2 p^n$

Numerical example: BLS12-446 bits

$p(x) = (x-1)^2(x^4 - x^2 + 1)/3 + x$
$r(x) = x^4 - x^2 + 1$
$s = -(2^{74} + 2^{73} + 2^{63} + 2^{57} + 2^{50} + 2^{17} + 1)$
seed with `enumerate_sparse_T.sage` [GMT20]
`https://gitlab.inria.fr/smasson/cocks-pinch-variant`
$p = p(s)$ of 446 bits, twist-secure curve
p^k 5352 bits
$h = Y^6 - Y^4 + Y^3 - Y + 1$

$f_y = X^{12} - 2yX^{10} + 2y^3X^6 + y^5X^2 + y^4 - y^3 + y - 1$
$g_y = X^2 - uy = X^2 + 28343567510342708887553y$
$A = 968,\ B = 2^{68.2}$

Estimated cost: $\approx 2^{132}$

Differences

- Barbulescu–Duquesne [BD19] (curve name, prime field GF(p) bitzise):
  - BN-462 (p^{12}: 5544 bits), BLS12-461 (p^{12}: 5532 bits) for the 128-bit security level
  - BLS24-559 (p^{24} 13416 bits) for the 192-bit security level
- Guillevic–Singh [GS21]:
  - BN-446, BLS12-446 (p^{12} 5352 bits), 64-bit machine-word aligned
  - BLS24-509 (p^{12} 12216 bits)

Differences

- Barbulescu–Duquesne [BD19] (curve name, prime field GF(p) bitzise):
  - BN-462 (p^{12}: 5544 bits), BLS12-461 (p^{12}: 5532 bits) for the 128-bit security level
  - BLS24-559 (p^{24} 13416 bits) for the 192-bit security level
- Guillevic–Singh [GS21]:
  - BN-446, BLS12-446 (p^{12} 5352 bits), 64-bit machine-word aligned
  - BLS24-509 (p^{12} 12216 bits)
- shorter p bitsize, one 64-bit machine-word less $\to$ faster $\mathbb{F}_p$-multiplication, ratio of $(2s^2 + s)/(2s_0^2 + s_0)$, $s = \lceil p/64 \rceil$ [AFK$^+$13, Sect. 8]
  462-bit $\to$ 446-bit: $\mathbf{m}_{446} = 0.77\mathbf{m}_{462}$
  559-bit $\to$ 509-bit: $\mathbf{m}_{509} = 0.8\mathbf{m}_{559}$
- faster pairing, faster group operations, shorter keysizes

Differences

Keysize recommendation difference:
[BD19] assumes there exists *optimal* polynomial h
and the attacker knows how to select it

BLS24

There exists $h(y)$ of degree 24 such that

- $\|h\|_\infty = 1$ i.e. $h_i \in \{0, 1, -1\}$
- h irreducible mod p of a BLS24 curve
- h has cyclic Galois group of order 24

Open problem: *Does it exist? How to find such $h(y)$?*
Ideas are welcome

Ongoing work

Active branches

automorphisms

fab46aea · taking into account special automorphisms for cyclotomic polynomials h. Tested... · 3 weeks ago

master default protected

378f61dd · comment on BLS24 seeds · 1 month ago

Ongoing work

Finding curve seeds of low Hamming weight

```
sage -python -m tnfs.gen.generate_sparse_curve --bls \
    -k 24 -r 254 256 --2NAF --find_all_w_up_to -w 4
cat \
test_vector_sparse_bls24_rnbits_254_256_u_1_4_mod_6_unbits_33_Hw2naf_6.py
test_vector_sparse_bls24 = [
    {'u':-0xeffff000, ... 'label':"-2^32+2^28+2^12 Hw2naf 3"},
```

With high 2-valuation of $p-1$ and $r-1$ for Youssef El Housni

```
sage -python -m tnfs.gen.compute_test_vector_curve --bls \
    -k 24 -r 254 256 --find_all_u --valuation 16
cat \
test_vector_bls24_rnbits_254_256_val2_16_r_prime_pos_u__u_1_4_mod_6.py
# BLS24 curves with seed u = [1, 4] mod 6 s.t. r has 254 to 256 bits
test_vector_BLS24 = [
  {'u':0xe19c0001, 'u_mod_4':1, 'b': 1, 'pnbits':317,'rnbits':255, \
```

Thank you.

https://gitlab.inria.fr/tnfs-alpha/alpha

Bibliography I

Diego F. Aranha, Laura Fuentes-Castañeda, Edward Knapp, Alfred Menezes, and Francisco Rodríguez-Henríquez.
Implementing pairings at the 192-bit security level.
In Michel Abdalla and Tanja Lange, editors, *PAIRING 2012*, volume 7708 of *LNCS*, pages 177–195. Springer, Heidelberg, May 2013.

Razvan Barbulescu and Sylvain Duquesne.
Updating key size estimations for pairings.
Journal of Cryptology, 32(4):1298–1336, October 2019.

Fabrice Boudot, Pierrick Gaudry, Aurore Guillevic, Nadia Heninger, Emmanuel Thomé, and Paul Zimmermann.
Comparing the difficulty of factorization and discrete logarithm: A 240-digit experiment.
In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 62–91. Springer, Heidelberg, August 2020.

Razvan Barbulescu, Pierrick Gaudry, Aurore Guillevic, and François Morain.
Improving NFS for the discrete logarithm problem in non-prime finite fields.
In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 129–155. Springer, Heidelberg, April 2015.

Bibliography II

Razvan Barbulescu, Pierrick Gaudry, and Thorsten Kleinjung.
The tower number field sieve.
In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 31–55. Springer, Heidelberg, November / December 2015.

Gabrielle De Micheli, Pierrick Gaudry, and Cécile Pierrot.
Lattice enumeration for tower NFS: A 521-bit discrete logarithm computation.
In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part I*, volume 13090 of *LNCS*, pages 67–96. Springer, Heidelberg, December 2021.

Aurore Guillevic, Simon Masson, and Emmanuel Thomé.
Cocks–Pinch curves of embedding degrees five to eight and optimal ate pairing computation.
Des. Codes Cryptography, 88:1047–1081, March 2020.

Aurore Guillevic and Shashank Singh.
On the alpha value of polynomials in the tower number field sieve algorithm.
Mathematical Cryptology, 1(1):1–39, Feb. 2021.

Antoine Joux and Cécile Pierrot.
The special number field sieve in $\mathbb{F}_{p^n}$ - application to pairing-friendly constructions.
In Zhenfu Cao and Fangguo Zhang, editors, *PAIRING 2013*, volume 8365 of *LNCS*, pages 45–61. Springer, Heidelberg, November 2014.

Bibliography III

Taechan Kim and Razvan Barbulescu.
Extended tower number field sieve: A new complexity for the medium prime case.
In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 543–571. Springer, Heidelberg, August 2016.

Taechan Kim and Jinhyuck Jeong.
Extended tower number field sieve with application to finite fields of arbitrary composite extension degree.
In Serge Fehr, editor, *PKC 2017, Part I*, volume 10174 of *LNCS*, pages 388–408. Springer, Heidelberg, March 2017.

Palash Sarkar and Shashank Singh.
A general polynomial selection method and new asymptotic complexities for the tower number field sieve algorithm.
In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 37–62. Springer, Heidelberg, December 2016.