On the Function Field Sieve and the Impact of Higher Splitting Probabilities

Robert Granger
robbiegranger@gmail.com

Joint work with Faruk Göloğlu, Gary McGuire and Jens Zumbrägel

Claude Shannon Institute
Complex & Adaptive Systems Laboratory
School of Mathematical Sciences
University College Dublin, Ireland

16th September, ECC 2013
1 Cracks in the Armour
  - Polynomial Time Relation Generation
  - Polynomial Time Degree 2 Logarithms
  - Solving the DLP in $\mathbb{F}_{2^{1971}}$ and $\mathbb{F}_{2^{3164}}$
1. **Cracks in the Armour**
   - Polynomial Time Relation Generation
   - Polynomial Time Degree 2 Logarithms
   - Solving the DLP in $\mathbb{F}_{21971}$ and $\mathbb{F}_{23164}$

2. **What Armour?**
   - Big Field Hunting
   - Solving the DLP in $\mathbb{F}_{26120}$
Overview

1. Cracks in the Armour
   - Polynomial Time Relation Generation
   - Polynomial Time Degree 2 Logarithms
   - Solving the DLP in $\mathbb{F}_{2^{1971}}$ and $\mathbb{F}_{2^{3164}}$

2. What Armour?
   - Big Field Hunting
   - Solving the DLP in $\mathbb{F}_{2^{6120}}$

3. Final Remarks
Consider the DLP in $\mathbb{F}_{q^n}$. The ICM consists of two stages:
Consider the DLP in $\mathbb{F}_{q^n}$. The ICM consists of two stages:

1. Choose a factor base $\mathcal{F}$, find relations between elements and then compute their logarithms.
Consider the DLP in $\mathbb{F}_{q^n}$. The ICM consists of two stages:

1. Choose a factor base $\mathcal{F}$, find relations between elements and then compute their logarithms.
2. For an arbitrary element, express it as a product of lower degree elements; recurse until all leaves are in $\mathcal{F}$. 
The Index Calculus Method

Consider the DLP in $\mathbb{F}_{q^n}$. The ICM consists of two stages:

1. Choose a factor base $\mathcal{F}$, find relations between elements and then compute their logarithms.

2. For an arbitrary element, express it as a product of lower degree elements; recurse until all leaves are in $\mathcal{F}$. 
The Joux-Lercier FFS variation [JL06]

To find factor base relations in \( \mathbb{F}_{q^n} \) one uses the following setup.
The Joux-Lercier FFS variation [JL06]

To find factor base relations in $\mathbb{F}_{q^n}$ one uses the following setup.

- Choose $g_1, g_2 \in \mathbb{F}_q[X]$ of degrees $d_1, d_2$ such that $X - g_1(g_2(X))$ has a degree $n$ irreducible factor $f(X)$ over $\mathbb{F}_q$, then $\mathbb{F}_{q^n} = \mathbb{F}_q(x) \cong \mathbb{F}_q[X]/(f(X)\mathbb{F}_q[X])$
- Let $y = g_2(x)$; then $x = g_1(y)$ and $\mathbb{F}_{q^n} \cong \mathbb{F}_q(x) \cong \mathbb{F}_q(y)$
- In best case factor base is $\{x - a \mid a \in \mathbb{F}_q\} \cup \{y - b \mid b \in \mathbb{F}_q\}$
The Joux-Lercier FFS variation [JL06]

To find factor base relations in \( \mathbb{F}_{q^n} \) one uses the following setup.

- Choose \( g_1, g_2 \in \mathbb{F}_q[X] \) of degrees \( d_1, d_2 \) such that \( X - g_1(g_2(X)) \) has a degree \( n \) irreducible factor \( f(X) \) over \( \mathbb{F}_q \), then \( \mathbb{F}_{q^n} = \mathbb{F}_q(x) \cong \mathbb{F}_q[X]/(f(X)\mathbb{F}_q[X]) \)
- Let \( y = g_2(x) \); then \( x = g_1(y) \) and \( \mathbb{F}_{q^n} \cong \mathbb{F}_q(x) \cong \mathbb{F}_q(y) \)
- In best case factor base is \( \{x - a \mid a \in \mathbb{F}_q\} \cup \{y - b \mid b \in \mathbb{F}_q\} \)

Relation generation:
The Joux-Lercier FFS variation [JL06]

To find factor base relations in $\mathbb{F}_{q^n}$ one uses the following setup.

- Choose $g_1, g_2 \in \mathbb{F}_q[X]$ of degrees $d_1, d_2$ such that $X - g_1(g_2(X))$ has a degree $n$ irreducible factor $f(X)$ over $\mathbb{F}_q$, then $\mathbb{F}_{q^n} = \mathbb{F}_q(x) \cong \mathbb{F}_q[X]/(f(X)\mathbb{F}_q[X])$.

- Let $y = g_2(x)$; then $x = g_1(y)$ and $\mathbb{F}_{q^n} \cong \mathbb{F}_q(x) \cong \mathbb{F}_q(y)$.

- In best case factor base is $\{x - a \mid a \in \mathbb{F}_q\} \cup \{y - b \mid b \in \mathbb{F}_q\}$

Relation generation:

- Considering elements $xy + ay + bx + c$ with $a, b, c \in \mathbb{F}_q$, one obtains the $\mathbb{F}_{q^n}$-equality

$$xg_2(x) + ag_2(x) + bx + c = yg_1(y) + ay + bg_1(y) + c$$

- When both sides split over $\mathbb{F}_q$ one obtains a relation
Optimising $d_1$ and $d_2$ in [JL06]

Fundamental Theorem of Cryptography

“If we have no clue about something, then we can safely assume that it behaves as a uniformly distributed random variable.”

– Igor Shparlinski
Optimising $d_1$ and $d_2$ in [JL06]

Fundamental Theorem of Cryptography

“If we have no clue about something, then we can safely assume that it behaves as a uniformly distributed random variable.”

– Igor Shparlinski

F.T.C. $\iff$ that as $q \to \infty$ each side of $xy + ay + bx + c$ splits over $\mathbb{F}_q$ with probability $1/(d_2 + 1)!$ and $1/(d_1 + 1)!$ respectively.
Optimising $d_1$ and $d_2$ in [JL06]

Fundamental Theorem of Cryptography

“If we have no clue about something, then we can safely assume that it behaves as a uniformly distributed random variable.”

– Igor Shparlinski

F.T.C. $\implies$ that as $q \to \infty$ each side of $xy + ay + bx + c$ splits over $\mathbb{F}_q$ with probability $1/(d_2 + 1)!$ and $1/(d_1 + 1)!$ respectively.

$\implies$ Choose $d_1 \approx d_2 \approx \sqrt{n}$
Fundamental Theorem of Cryptography

"If we have no clue about something, then we can safely assume that it behaves as a uniformly distributed random variable."

— Igor Shparlinski

F.T.C. $\iff$ that as $q \to \infty$ each side of $xy + ay + bx + c$ splits over $\mathbb{F}_q$ with probability $1/(d_2 + 1)!$ and $1/(d_1 + 1)!$ respectively.

- Choose $d_1 \approx d_2 \approx \sqrt{n}$
- For $q = L_q^*(1/3, 3^{-2/3})$ algorithm is $L_q^*(1/3, 3^{1/3})$
Optimising $d_1$ and $d_2$ in [JL06]

Fundamental Theorem of Cryptography

“If we have no clue about something, then we can safely assume that it behaves as a uniformly distributed random variable.”

— Igor Shparlinski

F.T.C. $\implies$ that as $q \to \infty$ each side of $xy + ay + bx + c$ splits over $\mathbb{F}_q$ with probability $1/(d_2 + 1)!$ and $1/(d_1 + 1)!$ respectively.

- $\implies$ Choose $d_1 \approx d_2 \approx \sqrt{n}$
- For $q = L_q^n(1/3, 3^{-2/3})$ algorithm is $L_q^n(1/3, 3^{1/3})$

A Counterpoint to the F.T.C.

Fortunately, in one sub-case of the [JL06] setup, we do have a clue.
An auspicious choice for $g_2$

Assume now that the base field is $\mathbb{F}_{q^k}$ for $k \geq 2$
An auspicious choice for $g_2$

Assume now that the base field is $\mathbb{F}_{q^k}$ for $k \geq 2$

- Let $y = g_2(x) = x^q$
An auspicious choice for $g_2$

Assume now that the base field is $\mathbb{F}_{q^k}$ for $k \geq 2$

- Let $y = g_2(x) = x^q$
- Eliminates half of the factor base since

$$(y + b) = (x + b^{1/q})^q \implies \log(y + b) = q \log(x + b^{1/q})$$
An auspicious choice for $g_2$

Assume now that the base field is $\mathbb{F}_{q^k}$ for $k \geq 2$

- Let $y = g_2(x) = x^q$
- Eliminates half of the factor base since
  \[(y + b) = (x + b^{1/q})^q \implies \log(y + b) = q \log(x + b^{1/q})\]
- The l.h.s. of $xy + ay + bx + c$ becomes
  \[x^{q+1} + ax^q + bx + c\]
An auspicious choice for $g_2$

Assume now that the base field is $\mathbb{F}_{q^k}$ for $k \geq 2$

- Let $y = g_2(x) = x^q$
- Eliminates half of the factor base since

\[(y + b) = (x + b^{1/q})^q \implies \log(y + b) = q \log(x + b^{1/q})\]

- The l.h.s. of $xy + ay + bx + c$ becomes

\[x^{q+1} + ax^q + bx + c\]

- This polynomial \textit{provably} splits over $\mathbb{F}_{q^k}$ with probability

\[\approx 1/q^3 \gg 1/(q + 1)!\]
Bluher polynomials

Let \( k \geq 3 \) and consider the polynomial \( X^{q+1} + aX^q + bX + c \).
Bluher polynomials

Let $k \geq 3$ and consider the polynomial $X^{q+1} + aX^q + bX + c$.

If $ab \neq c$ and $a^q \neq b$, this may be transformed into

$$F_B(X) = X^{q+1} + BX + B,$$

with $B = \frac{(b - a^q)^{q+1}}{(c - ab)^q}$,

via $X = \frac{c - ab}{b - a^q} X - a$. 
Bluher polynomials

Let \( k \geq 3 \) and consider the polynomial \( X^{q+1} + aX^q + bX + c \).

If \( ab \neq c \) and \( a^q \neq b \), this may be transformed into

\[
F_B(X) = X^{q+1} + B X + B, \quad \text{with} \quad B = \frac{(b - a^q)^{q+1}}{(c - ab)^q},
\]

via \( X = \frac{c - ab}{b - a^q} \overline{X} - a \).

**Theorem (Bluher 2004)**

The number of elements \( B \in \mathbb{F}_q^\times \) such that the polynomial \( F_B(X) \in \mathbb{F}_q^k[X] \) splits completely over \( \mathbb{F}_q^k \) equals

\[
\frac{q^{k-1} - 1}{q^2 - 1} \quad \text{if } k \text{ is odd}, \quad \frac{q^{k-1} - q}{q^2 - 1} \quad \text{if } k \text{ is even}.
\]
Polynomial time relation generation: \( k \geq 3 \)

Assume that \( g_1 \) can be found s.t. \( X - g_1(X^q) \equiv 0 \pmod{f(X)} \) with \( \deg(f) = n \leq qd_1 \). Then we have the following method:
Polynomial time relation generation: \( k \geq 3 \)

Assume that \( g_1 \) can be found s.t. \( X - g_1(X^q) \equiv 0 \pmod{f(X)} \) with \( \deg(f) = n \leq qd_1 \). Then we have the following method:

- Compute \( S_B = \{ B \in \mathbb{F}_{q^k}^\times \mid X^{q+1} + BX + B \text{ splits over } \mathbb{F}_{q^k} \} \)
Polynomial time relation generation: \( k \geq 3 \)

Assume that \( g_1 \) can be found s.t. \( X - g_1(X^q) \equiv 0 \pmod{f(X)} \) with \( \deg(f) = n \leq qd_1 \). Then we have the following method:

- Compute \( S_B = \{ B \in \mathbb{F}_{q^k}^\times \mid X^{q+1} + BX + B \text{ splits over } \mathbb{F}_{q^k} \} \)
- Since \( B = (b - a^q)^{q+1}/(c - ab)^q \), for any \( a, b \in \mathbb{F}_{q^k} \) s.t. \( b \neq a^q \), and \( B \in S_B \), there exists a unique \( c \in \mathbb{F}_{q^k} \) s.t. \( x^{q+1} + ax^q + bx + c \) splits over \( \mathbb{F}_{q^k} \)
Polynomial time relation generation: $k \geq 3$

Assume that $g_1$ can be found s.t. $X - g_1(X^q) \equiv 0 \pmod{f(X)}$ with $\deg(f) = n \leq qd_1$. Then we have the following method:

- Compute $S_B = \{ B \in \mathbb{F}_{qk}^\times \mid X^{q+1} + BX + B \text{ splits over } \mathbb{F}_{qk} \}$

- Since $B = (b - a^q)q^{q+1}/(c - ab)^q$, for any $a, b \in \mathbb{F}_{qk}$ s.t. $b \neq a^q$, and $B \in S_B$, there exists a unique $c \in \mathbb{F}_{qk}$ s.t. $x^{q+1} + ax^q + bx + c$ splits over $\mathbb{F}_{qk}$

- For each such $(a, b, c)$, test if r.h.s. $yg_1(y) + ay + bg_1(y) + c$ splits; if so then have a relation
Polynomial time relation generation: \( k \geq 3 \)

Assume that \( g_1 \) can be found s.t. \( X - g_1(X^q) \equiv 0 \pmod{f(X)} \) with \( \deg(f) = n \leq qd_1 \). Then we have the following method:

- Compute \( S_B = \{ B \in \mathbb{F}_q^x \mid X^{q+1} + BX + B \text{ splits over } \mathbb{F}_q^k \} \)
- Since \( B = (b - a^q)^{q+1}/(c - ab)^q \), for any \( a, b \in \mathbb{F}_q^k \) s.t. \( b \neq a^q \), and \( B \in S_B \), there exists a unique \( c \in \mathbb{F}_q^k \) s.t. \( x^{q+1} + ax^q + bx + c \text{ splits over } \mathbb{F}_q^k \)
- For each such \( (a, b, c) \), test if r.h.s. \( yg_1(y) + ay + bg_1(y) + c \) splits; if so then have a relation
- If \( q^{3k-3} > q^k(d_1 + 1)! \) then for \( d_1 \geq 1 \) constant we expect to compute logs of degree 1 elements of \( \mathbb{F}_{(q^k)^n} \) in time

\[ O(q^{2k+1}) \]
Polynomial time relation generation: \( k = 2 \)

For the base field \( \mathbb{F}_{q^2} \), relevant set of triples is

\[
\{(a, a^q, c) \mid a \in \mathbb{F}_{q^2} \text{ and } c \in \mathbb{F}_q, c \neq a^{q+1}\}.
\]

- Hence \( q^2(q - 1) = q^3 - q^2 \) polys \( x^{q+1} + ax^q + a^qx + c \)
- For each \((a, a^q, c)\), test if r.h.s. splits; if so then a relation
- If \( q^3 - q^2 > q^2(d_1 + 1)! \) then for \( d_1 \geq 1 \) constant we expect to compute logs of degree 1 elements of \( \mathbb{F}_{(q^2)^n} \) in time

\[ O(q^5) \]
Polynomial time relation generation - examples

Let $q = 2^l$, let the base field be $\mathbb{F}_{q^3}$ and let $n = q - 1$.

- Since $n \mid (q^3 - 1)$ one can use a Kummer extension
- Set $g_1(X) = \gamma X$, so that irreducible is $X^{q-1} + \gamma$
- r.h.s. has degree 2 and splits with probability 1/2
Let $q = 2^l$, let the base field be $\mathbb{F}_{q^3}$ and let $n = q - 1$.

- Since $n | (q^3 - 1)$ one can use a Kummer extension
- Set $g_1(X) = \gamma X$, so that irreducible is $X^{q-1} + \gamma$
- r.h.s. has degree 2 and splits with probability $1/2$

**Table:** Timings for $\mathbb{F}_{q^{3n}}$ on a 2.0GHz AMD Opteron 6128

<table>
<thead>
<tr>
<th>$l$</th>
<th>$\log_2(q^{3n})$</th>
<th>#vars</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>2667</td>
<td>5506</td>
<td>2.3s</td>
</tr>
<tr>
<td>8</td>
<td>6120</td>
<td>21932</td>
<td>15.0s</td>
</tr>
<tr>
<td>9</td>
<td>13797</td>
<td>87554</td>
<td>122s</td>
</tr>
<tr>
<td>10</td>
<td>30690</td>
<td>349858</td>
<td>900s</td>
</tr>
</tbody>
</table>
The Index Calculus Method (reminder)

Consider the DLP in $\mathbb{F}_{q^n}$. The ICM consists of two stages:

1. Choose a factor base $\mathcal{F}$, find relations between elements and then compute their logarithms.

2. For an arbitrary element, express it as a product of lower degree elements; recurse until all leaves are in $\mathcal{F}$. 
Consider the DLP in $\mathbb{F}_{q^n}$. The ICM consists of two stages:

1. Choose a factor base $\mathcal{F}$, find relations between elements and then compute their logarithms.

2. For an arbitrary element, express it as a product of lower degree elements; recurse until all leaves are in $\mathcal{F}$. 
Special- $Q$ elimination

Let $Q(x) \in \mathbb{F}_{(q^k)^n}$ be a degree $D$ element which is to be written as a product of lower degree elements. Since $y = x^q$ we have

$$Q(x)^q = \overline{Q}(x^q) = \overline{Q}(y),$$

where the coefficients of $\overline{Q}$ are those of $Q$, powered by $q$.

- We use the special- $\overline{Q}$ lattice $L_{\overline{Q}}$ defined by

$$L_{\overline{Q}} = \{(w_0, w_1) \in \mathbb{F}_{q^k}[Y]^2 \mid w_0 g_1 + w_1 \equiv 0 \quad (\text{mod } \overline{Q})\}$$

- A basis for $L_{\overline{Q}}$ is $(0, \overline{Q}), (1, -g_1 \mod \overline{Q})$
- A reduced basis $(u_0, u_1), (v_0, v_1)$ exists with degrees $\approx D/2$
- For $r, s \in \mathbb{F}_{q^k}[Y]$, we have $(w_0, w_1) \in L_{\overline{Q}}$ where

$$(w_0, w_1) = (ru_0 + sv_0, ru_1 + sv_1)$$
Special- $Q$ elimination

Recall that in $\mathbb{F}_{(q^k)^n}$ we have $y = x^q$ and $x = g_1(y)$. Therefore:

$$w_0(x^q)x + w_1(x^q) = w_0(y)g_1(y) + w_1(y),$$

where the r.h.s. is divisible by $\overline{Q}(y)$ for all $r, s$.

- Search over small degree $r, s$ until l.h.s. and r.h.s./$\overline{Q}(y)$ are both $(D - 1)$-smooth $\implies$ elimination of $\overline{Q}$ is complete
Degree 2 logarithms

Let $\overline{Q}(y) = y^2 + \overline{q}_1 y + \overline{q}_0 \in \mathbb{F}_{(q^k)^n}$ be an element to be eliminated, i.e., written as a product of linear elements.

- $L_{\overline{Q}}$ has reduced basis $(u_0, u_1), (v_0, v_1)$ of max. degree 1
- For $s \in \mathbb{F}_{q^k}$, we have $(w_0, w_1)$ each of degree 1 where

$$ (w_0, w_1) = (u_0 + s v_0, u_1 + s v_1) \in L_{\overline{Q}} $$

- r.h.s. $(u_0(y) + sv_0(y))g_1(y) + (u_1(y) + sv_1(y))$ has degree $d_1 + 1$, so cofactor splits with probability $1/(d_1 - 1)!$
Degree 2 logarithms

Let $\overline{Q}(y) = y^2 + \overline{q}_1 y + \overline{q}_0 \in \mathbb{F}_{q^k}^n$ be an element to be eliminated, i.e., written as a product of linear elements.

- $L_{\overline{Q}}$ has reduced basis $(u_0, u_1), (v_0, v_1)$ of max. degree 1
- For $s \in \mathbb{F}_{q^k}$, we have $(w_0, w_1)$ each of degree 1 where

$$ (w_0, w_1) = (u_0 + s v_0, u_1 + s v_1) \in L_{\overline{Q}} $$

- r.h.s. $(u_0(y) + sv_0(y))g_1(y) + (u_1(y) + sv_1(y))$ has degree $d_1 + 1$, so cofactor splits with probability $1/(d_1 - 1)!$
- Corresponding l.h.s. is $w_0(x^q) x + w_1(x^q)$, which is of the form

$$ x^{q+1} + ax^q + bx + c $$

which therefore splits with high probability ($\approx 1/q^3$)
Degree 2 logarithms

- Write the basis of $L_Q$ as $(Y + u_{00}, u_{10}), (v_{00}, Y + v_{10})$
Degree 2 logarithms

- Write the basis of $L_Q$ as $(Y + u_{00}, u_{10}), (v_{00}, Y + v_{10})$
- L.h.s. is $x^{q+1} + sx^q + (u_{00} + sv_{00})x + (u_{10} + sv_{10})$
Degree 2 logarithms

- Write the basis of $L_Q$ as $(Y + u_{00}, u_{10}), (v_{00}, Y + v_{10})$
- I.h.s. is $x^{q+1} + sx^q + (u_{00} + sv_{00})x + (u_{10} + sv_{10})$
- For each $B \in S_B$ we try to solve $B = (b - a^q)^{q+1}/(c - ab)^q$ for $s$, i.e., find $s \in \mathbb{F}_{q^k}$ that satisfies the $\mathbb{F}_{q^k}[S]$ polynomial

\[
B \cdot (u_{10} + v_{10}S - (u_{00}S + v_{00}S^2))^q = (u_{00} + v_{00}S - S^q)^{q+1},
\]

via GCD with $S^{q^k} - S$: Cost is $O(q^2 \log q^k)$ $\mathbb{F}_{q^k}$-ops
Degree 2 logarithms

- Write the basis of $L_Q$ as $(Y + u_{00}, u_{10}), (v_{00}, Y + v_{10})$
- L.h.s. is $x^{q+1} + sx^q + (u_{00} + sv_{00})x + (u_{10} + sv_{10})$
- For each $B \in S_B$ we try to solve $B = (b - a^q)^{q+1}/(c - ab)^q$
  for $s$, i.e., find $s \in \mathbb{F}_{q^k}$ that satisfies the $\mathbb{F}_{q^k}[S]$ polynomial

$$B \cdot (u_{10} + v_{10}S - (u_{00}S + v_{00}S^2))^q = (u_{00} + v_{00}S - S^q)^{q+1},$$

via GCD with $S^{q^k} - S$: Cost is $O(q^2 \log q^k)$ \(\mathbb{F}_{q^k}\)-ops
- Probability of success is $\approx 1 - \left(1 - \frac{1}{2(d_1-1)!}\right)q^{k-3}$
Degree 2 logarithms

- Write the basis of $L_Q$ as $(Y + u_{00}, u_{10}), (v_{00}, Y + v_{10})$
- l.h.s. is $x^{q+1} + sx^{q} + (u_{00} + sv_{00})x + (u_{10} + sv_{10})$
- For each $B \in S_B$ we try to solve $B = (b - a^q)^{q+1}/(c - ab)^q$ for $s$, i.e., find $s \in \mathbb{F}_{q^k}$ that satisfies the $\mathbb{F}_{q^k}[S]$ polynomial

$$B \cdot (u_{10} + v_{10}S - (u_{00}S + v_{00}S^2))^q = (u_{00} + v_{00}S - S^q)^{q+1},$$

via GCD with $S^{q^k} - S$: Cost is $O(q^2 \log q^k)$ $\mathbb{F}_{q^k}$-ops
- Probability of success is $\approx 1 - \left(1 - \frac{1}{2(d_1 - 1)!}\right)^{q^{k-3}}$
- Hence need $q^{k-3} > 2(d_1 - 1)!$ to eliminate $\overline{Q}(y)$ with good probability: Expected cost is

$$O(q^2(d_1 - 1)! \log q^k) \mathbb{F}_{q^k}$-ops
Degree 2 logarithms (intelligently) [GGMZ13b]

Need to compute $s \in \mathbb{F}_{q^k}$ that satisfy the equation:

$$B \cdot (u_{10} + v_{10}s - (u_{00}s + v_{00}s^2))^q = (u_{00} + v_{00}s - s^q)^{q+1}$$
Need to compute $s \in \mathbb{F}_{q^k}$ that satisfy the equation:

$$B \cdot (u_{10} + v_{10}s - (u_{00}s + v_{00}s^2))^q = (u_{00} + v_{00}s - s^q)^{q+1}$$

- Use an explicit $\mathbb{F}_{q^k}/\mathbb{F}_q$ basis $\{1, \alpha, \ldots, \alpha^{k-1}\}$, and introduce $\mathbb{F}_q$-variables $s_0, \ldots, s_{k-1}$ s.t. $s = s_0 + s_1\alpha + \cdots + s_{k-1}\alpha^{k-1}$
Degree 2 logarithms (intelligently) [GGMZ13b]

Need to compute $s \in \mathbb{F}_{q^k}$ that satisfy the equation:

$$B \cdot (u_{10} + v_{10}s - (u_{00}s + v_{00}s^2))^q = (u_{00} + v_{00}s - s^q)^{q+1}$$

- Use an explicit $\mathbb{F}_{q^k}/\mathbb{F}_q$ basis $\{1, \alpha, \ldots, \alpha^{k-1}\}$, and introduce $\mathbb{F}_q$-variables $s_0, \ldots, s_{k-1}$ s.t. $s = s_0 + s_1 \alpha + \cdots + s_{k-1} \alpha^{k-1}$

- Gives a quadratic system, solvable in $O((k(2k_{k+1}))^{\omega})$ $\mathbb{F}_q$-ops
Degree 2 logarithms (intelligently) [GGMZ13b]

Need to compute $s \in \mathbb{F}_{q^k}$ that satisfy the equation:

$$B \cdot (u_{10} + v_{10}s - (u_{00}s + v_{00}s^2))^q = (u_{00} + v_{00}s - s^q)^{q+1}$$

- Use an explicit $\mathbb{F}_{q^k}/\mathbb{F}_q$ basis $\{1, \alpha, \ldots, \alpha^{k-1}\}$, and introduce $\mathbb{F}_q$-variables $s_0, \ldots, s_{k-1}$ s.t. $s = s_0 + s_1 \alpha + \cdots + s_{k-1} \alpha^{k-1}$
- Gives a quadratic system, solvable in $O((k \binom{2k}{k+1})^\omega)$ $\mathbb{F}_q$-ops
- For fixed $k$, $d_1$ and $q \to \infty$ this method has cost $O(1)$ $\mathbb{F}_q$-ops, i.e., it has polylogarithmic complexity
Complexity Results

Suppose \( q^k = \exp \left( \alpha \sqrt[3]{\log q^{kn} \cdot \log^2 \log q^{kn}} \right) \). Then we have:
Suppose $q^k = \exp \left( \alpha \sqrt[3]{\log q^{kn}} \cdot \log^2 \log q^{kn} \right)$. Then we have:

**Heuristic Result (i)**

For $n \approx q d_1$ and $q \approx d_1$, the DLP can be solved with complexity $L_{q^{kn}}(1/3, (8/9)^{1/3}) \approx L_Q(1/3, 0.961)$. 
Complexity Results

Suppose $q^k = \exp\left(\alpha \sqrt[3]{\log q^{kn}} \cdot \log^2 \log q^{kn}\right)$. Then we have:

**Heuristic Result (i)**

For $n \approx q d_1$ and $q \approx d_1$, the DLP can be solved with complexity $L_{q^{kn}}(1/3, (8/9)^{1/3}) \approx L_Q(1/3, 0.961)$.

**Heuristic Result (ii)**

Assume $\text{char}(\mathbb{F}_q) = 2$. Then for $n \approx q d_1$ and $q \gg d_1$, the DLP can be solved with complexity between

$$L_Q(1/3, (4/9)^{1/3}) \approx L_Q(1/3, 0.763) \quad \text{and}$$

$$L_Q(1/3, (1/2)^{1/3}) \approx L_Q(1/3, 0.794).$$
Solving the DLP in $\mathbb{F}_{2^{1971}}$

Let $q = 2^3$, $k = 9$ and $n = 73$. Our field representation was:

$$
\mathbb{F}_{q^9} = \mathbb{F}_{2^{27}} = \mathbb{F}_2[T]/(T^{27} + T^5 + T^2 + T + 1) = \mathbb{F}_2(t)
$$

$$
\mathbb{F}_{2^{1971}} = \mathbb{F}_{(q^9)^{73}} = \mathbb{F}_{q^9}[X]/(X^{73} + t) = \mathbb{F}_{q^9}(x)
$$

Let $y = x^8$ and thus $x = t/y^9$. We took as (a presumed) generator $g = x + 1$ and target element

$$
h_\pi = \sum_{i=0}^{72} \tau(\lfloor \pi q^9(i+1) \rfloor \mod q^9) x^i.
$$
Solving the DLP in $\mathbb{F}_{2^{1971}}$

The computation took:

- **14 core-hrs** for relation generation: quotienting out by the action of the 9-th power of the $\mathbb{F}_2$-Frobenius on the factor base gives $612,872 \approx 2^{27}/(3 \cdot 73)$ variables

- After S.G.E., **2220 core-hrs** for parallelised Lanczos on matrix of dimension $528,812 \times 527,766$

- **898 core-hrs** for the descent $\implies$ total of **3132 core-hrs**
Solving the DLP in $\mathbb{F}_{2^{1971}}$

On 19/2/13 we announced that $\log_g(h_\pi) =$

\begin{align*}
11992984215354106866091146371988855845186852755447163352 \\
36895900760902198795745784008181148775933944656038305197 \\
82541742360236535889937362200771117361678269423101163403 \\
13535552228080411390321527355590590108228224824002192878 \\
78207304028565280573096588688279004416835100344085961912 \\
42700060128986433752110002214380289887546061125224587971 \\
19787275080584651962314043764573936293823541736161168108 \\
25627780459657892709561158924173579400674739684346062992 \\
68294291957378226451182620783745349502502960139927453196 \\
48974006524479548958327920827882768332440907342446643941 \\
0976702162039539513377673115483439
\end{align*}
Solving the DLP in $\mathbb{F}_{2^{3164}}$

Let $q = 2^4$, $k = 7$ and $n = 113$. Our field representation was:

$$\mathbb{F}_{q^7} = \mathbb{F}_{2^{28}} = \mathbb{F}_2[T]/(T^{28} + T + 1) = \mathbb{F}_2(t)$$
$$\mathbb{F}_{2^{3164}} = \mathbb{F}_{(q^7)^{113}} = \mathbb{F}_{q^7}[X]/(X^{113} + t) = \mathbb{F}_{q^7}(x)$$

Let $y = x^{16}$ and thus $x = t/y^7$. We took as (a proven) generator $g = x + t + 1$ and target element

$$h_π = \sum_{i=0}^{112} τ(\lfloor π q^7(i+1) \rfloor \mod q^7) x^i.$$
Solving the DLP in $\mathbb{F}_{2^{3164}}$

The computation took:

- **2 core-hrs** for relation generation: quotienting out by the action of the 14-th power of the $\mathbb{F}_2$-Frobenius on the factor base gives $1,187,841 \approx 2^{28}/(2 \cdot 113)$ variables

- After S.G.E., **85,488 core-hrs** for parallelised Lanczos on matrix of dimension $1,066,010 \times 1,064,991$

- **21,602 core-hrs** for the descent $\implies$ total of **107,092 core-hrs**
Solving the DLP in $\mathbb{F}_{2^{3164}}$

On 3/5/13 we found that $\log_g(h_\pi) =$

\[
2410958672084703779901202077261642209070514313288787533385808717024 \\
8784565712688312063491036765323357553857177477977665457317849564770 \\
1688094481773173140524389502529386852264636049383546885561763318178 \\
6341747893370309598402582718996263618673697554067799885512742832012 \\
3901294838991530024173934004391610582283400289720429303619769406533 \\
7903255793451858773664350130030722091666253172541070447948299781221 \\
0193428607010640365444303319677531146464806350633002030742348610674 \\
7166841199820454431917683235380198222192499580429542616711230697079 \\
596079898864631100037393291558580412406942004555116148790387654960 \\
4900084297695444007900819088072394071341577241660482464194055035573 \\
9803589799985259319695403143962976877685099988772087056174191305553 \\
1864041654707840433795403753200520891617150254756586728215941551355 \\
0648407797656823989931563900000242491107399569193500692930336704230 \\
7029958155763666499372120453686303873671488016409635578117870889230 \\
278649164378133
\]
Big Field Hunting 2013

- 9th Apr’13, CARAMEL: $\mathbb{F}_{2809}$ in < 20,000 core-hrs
- 11th Feb’13, Joux: $\mathbb{F}_{21778}$ in 220 core-hrs
- 19th Feb’13, GGMZ: $\mathbb{F}_{21971}$ in 3,132 core-hrs
- 3rd May’13, GGMZ: $\mathbb{F}_{23164}$ in 107,000 core-hrs
- 22nd Mar’13, Joux: $\mathbb{F}_{24080}$ in 14,100 core-hrs
- 11th Apr’13, GGMZ: $\mathbb{F}_{26120}$ in 750 core-hrs
- 21st May’13, Joux: $\mathbb{F}_{26168}$ in 550 core-hrs
Big Field Hunting 2013

- 9th Apr’13, CARAMEL: $\mathbb{F}_{2^{809}}$ in < 20,000 core-hrs
- 11th Feb’13, Joux: $\mathbb{F}_{2^{1778}}$ in 220 core-hrs
- 19th Feb’13, GGMZ: $\mathbb{F}_{2^{1971}}$ in 3,132 core-hrs
- 3rd May’13, GGMZ: $\mathbb{F}_{2^{3164}}$ in 107,000 core-hrs
- 22nd Mar’13, Joux: $\mathbb{F}_{2^{4080}}$ in 14,100 core-hrs
- 11th Apr’13, GGMZ: $\mathbb{F}_{2^{6120}}$ in 750 core-hrs
- 21st May’13, Joux: $\mathbb{F}_{2^{6168}}$ in 550 core-hrs
Big Field Hunting 2013

- 9th Apr’13, CARAMEL: $\mathbb{F}_{2^{809}}$ in $< 20,000$ core-hrs
- 11th Feb’13, Joux: $\mathbb{F}_{2^{1778}}$ in 220 core-hrs
- 19th Feb’13, GGMZ: $\mathbb{F}_{2^{1971}}$ in 3,132 core-hrs
- 3rd May’13, GGMZ: $\mathbb{F}_{2^{3164}}$ in 107,000 core-hrs
- 22nd Mar’13, Joux: $\mathbb{F}_{2^{4080}}$ in 14,100 core-hrs
- 11th Apr’13, GGMZ: $\mathbb{F}_{2^{6120}}$ in 750 core-hrs
- 21st May’13, Joux: $\mathbb{F}_{2^{6128}}$ in 550 core-hrs
Big Field Hunting 2013

- 9th Apr’13, CARAMEL: $\mathbb{F}_{2^{809}}$ in $< 20,000$ core-hrs
- 11th Feb’13, Joux: $\mathbb{F}_{2^{1778}}$ in 220 core-hrs
- 19th Feb’13, GGMZ: $\mathbb{F}_{2^{1971}}$ in 3,132 core-hrs
- 3rd May’13, GGMZ: $\mathbb{F}_{2^{3164}}$ in 107,000 core-hrs
- 22nd Mar’13, Joux: $\mathbb{F}_{2^{4080}}$ in 14,100 core-hrs
- 11th Apr’13, GGMZ: $\mathbb{F}_{2^{6120}}$ in 750 core-hrs
- 21st May’13, Joux: $\mathbb{F}_{2^{6168}}$ in 550 core-hrs
Kummer extensions \implies more efficient attacks

The solution of DLPs in $\mathbb{F}_{p^{47}}, \mathbb{F}_{p^{57}}, \mathbb{F}_{21778}, \mathbb{F}_{21971}, \mathbb{F}_{23164}$ and $\mathbb{F}_{24080}$ all used Kummer extensions.

\textbf{Why?} Factor base-preserving automorphisms reduce effective size of factor base \implies relation finding & linear algebra become faster.
Kummer extensions \(\implies\) more efficient attacks

The solution of DLPs in \(\mathbb{F}_{p^{47}}, \mathbb{F}_{p^{57}}, \mathbb{F}_{2^{1778}}, \mathbb{F}_{2^{1971}}, \mathbb{F}_{2^{3164}}\) and \(\mathbb{F}_{2^{4080}}\) all used Kummer extensions.

**Why?** Factor base-preserving automorphisms reduce effective size of factor base \(\implies\) relation finding & linear algebra become faster.

Observe that \(\mathbb{F}_{2^{1778}}\) and \(\mathbb{F}_{2^{4080}}\) are of the form \(\mathbb{F}_{(q^2)^{q-1}}\), for which:

- Degree 1 log cost \(O(q^3)\) for K.e.’s, or \(O(q^5)\) otherwise
- Degree 2 log cost \(O(q^6)\) for K.e.’s, or \(O(q^7)\) otherwise
Kummer extensions $\implies$ more efficient attacks

The solution of DLPs in $\mathbb{F}_{p^{47}}, \mathbb{F}_{p^{57}}, \mathbb{F}_{2^{1778}}, \mathbb{F}_{2^{1971}}, \mathbb{F}_{2^{3164}}$ and $\mathbb{F}_{2^{4080}}$ all used Kummer extensions.

Why? Factor base-preserving automorphisms reduce effective size of factor base $\implies$ relation finding & linear algebra become faster.

Observe that $\mathbb{F}_{2^{1778}}$ and $\mathbb{F}_{2^{4080}}$ are of the form $\mathbb{F}_{(q^2)^{q-1}}$, for which:

- Degree 1 logs cost $O(q^3)$ for K.e.’s, or $O(q^5)$ otherwise
- Degree 2 logs cost $O(q^6)$ for K.e.’s, or $O(q^7)$ otherwise

However, we know that for $\mathbb{F}_{(q^k)^{q\pm1}}$ with $k \geq 4$ one can compute logs of degree 2 elements on the fly [GGMZ13a].
Cost of computing factor base logs for K.e.’s

For $q = 2^l$ and $n = q - 1$, $\mathbb{F}_{(q^k)^n}$ has bitlength:

<table>
<thead>
<tr>
<th>$l \setminus k$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>756</td>
<td>1134</td>
<td>1512</td>
<td>1890</td>
<td>2268</td>
</tr>
<tr>
<td>7</td>
<td>1778</td>
<td>2667</td>
<td>3556</td>
<td>4445</td>
<td>5334</td>
</tr>
<tr>
<td>8</td>
<td>4080</td>
<td>6120</td>
<td>8160</td>
<td>10200</td>
<td>12240</td>
</tr>
<tr>
<td>9</td>
<td>9198</td>
<td>13797</td>
<td>18396</td>
<td>22995</td>
<td>27594</td>
</tr>
</tbody>
</table>

- **Degree 1:** #variables $\approx q^{k-1}$ so for $k \geq 2$, cost is $O(q^{2k-1})$
- **Degree 2:** For $k = 2, 3$ cost is $O(q^{2k+2})$, and free for $k \geq 4$

<table>
<thead>
<tr>
<th>$k$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>$O(q^6)$</td>
<td>$O(q^8)$</td>
<td>$O(q^7)$</td>
<td>$O(q^9)$</td>
<td>$O(q^{11})$</td>
</tr>
</tbody>
</table>
Cost of computing factor base logs for K.e.'s

For \( q = 2^l \) and \( n = q - 1 \), \( \mathbb{F}_{(q^k)^n} \) has bitlength:

<table>
<thead>
<tr>
<th>( l \setminus k )</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>756</td>
<td>1134</td>
<td>1512</td>
<td>1890</td>
<td>2268</td>
</tr>
<tr>
<td>7</td>
<td>1778</td>
<td>2667</td>
<td>3556</td>
<td>4445</td>
<td>5334</td>
</tr>
<tr>
<td>8</td>
<td>4080</td>
<td>6120</td>
<td>8160</td>
<td>10200</td>
<td>12240</td>
</tr>
<tr>
<td>9</td>
<td>9198</td>
<td>13797</td>
<td>18396</td>
<td>22995</td>
<td>27594</td>
</tr>
</tbody>
</table>

- **Degree 1:** \#variables \( \approx q^{k-1} \) so for \( k \geq 2 \), cost is \( O(q^{2k-1}) \)
- **Degree 2:** For \( k = 2, 3 \) cost is \( O(q^{2k+2}) \), and free for \( k \geq 4 \)

<table>
<thead>
<tr>
<th>( k )</th>
<th>( 2 )</th>
<th>( 3 )</th>
<th>( 4 )</th>
<th>( 5 )</th>
<th>( 6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>( O(q^6) )</td>
<td>( O(q^5) )</td>
<td>( O(q^7) )</td>
<td>( O(q^9) )</td>
<td>( O(q^{11}) )</td>
</tr>
</tbody>
</table>
\( \mathbb{F}_{2^{6120}} \) field setup and target element

- Let \( \mathbb{F}_{2^8} = \mathbb{F}_2[T]/((T^8 + T^4 + T^3 + T + 1)\mathbb{F}_2[T]) = \mathbb{F}_2(t) \)
- Let \( \mathbb{F}_{2^{24}} = \mathbb{F}_{2^8}[W]/((W^3 + t)\mathbb{F}_{2^8}[W]) = \mathbb{F}_{2^8}(w) \)
- Let \( \mathbb{F}_{2^{6120}} = \mathbb{F}_{2^{24}}[X]/((X^{255} + w + 1)\mathbb{F}_{2^{24}}[X]) = \mathbb{F}_{2^{24}}(x) \)
- Our generator is \( g = x + w \), which has proven order \( 2^{6120} - 1 \)

Our target element \( h_\pi \) was derived as usual from the \( 2^{24} \)-ary expansion of \( \pi \).
Degree 1 logarithms

- Used the only Bluher polynomial for \( k = 3 \), namely \( X^{257} + X + 1 \) and our relation generation method.
- Via automorphisms, reduced the number of variables to 21,932 and obtained 22,932 relations in 15.0 seconds using C++/NTL on a 2.0 GHz AMD Opteron 6128.
- For linear algebra, took as modulus the product of the largest 35 prime factors of \( 2^{6120} - 1 \), which has bitlength 5121.
- Ran a parallelised C/GMP implementation of Lanczos’ algorithm on four of the Intel (Westmere) Xeon E5650 hex-core processors of ICHEC’s SGI Altix ICE 8200EX Stokes cluster, completed in 60.5 core-hrs (2.5 hrs wall time).
Degree 2 logarithms

Since there is only one Bluher polynomial for \( k = 3 \), elimination probability is \( 1/2 \).
Degree 2 logarithms

Since there is only one Bluher polynomial for $k = 3$, elimination probability is $1/2$.

- When it fails, exploit the fact that $6 \mid 24$ giving the 64 Bluher polynomials of the form $X^{65} + BX + B \ / \mathbb{F}_{2^{24}}$
Since there is only one Bluher polynomial for $k = 3$, elimination probability is $1/2$.

- When it fails, exploit the fact that $6 \mid 24$ giving the 64 Bluher polynomials of the form $X^{65} + BX + B \mod F_{24}$.
- Results in a probabilistic method to eliminate any given degree 2 element with probability $p = 1 - 6.3 \times 10^{-15}$.
- ⟹ probability that at least one degree 2 irreducible is not eliminable is $1 - p^{22} = 2.7 \times 10^{-8}$.
- Implemented in MAGMA V2.16-12 on a 2.0GHz AMD Opteron 6128: each took on average 0.03 seconds.
The descent

- Computed random $h_\pi g^i = z_0/z_1$ until $z_0, z_1$ both 27-smooth: 10 core-hrs
- Degree-balanced classical descent until all polys 6-smooth: 495 core-hrs
- For degrees 6, 5, 4, 3 used an analogue of Joux's method [J13], but with the Bluher polynomial $X^{257} + X + 1$ rather than $X^{256} + X$, then degree 2's using our method: 183 core-hrs
- Pollard rho for other factors plus linear algebra $\implies$ total of 749.5 core-hrs
On 11/4/13 we announced that $h_\pi = g^{\log_2}$, with $\log_2 =$

\begin{align*}
13858759363978692625475711281231710092363615038969923664959317045177002801271780222348940986175 \\
81360131441835074256363730624426814293234742725215981661269579281168254431109654042538793880859 \\
540411103523827071772178822939281873403345199731815140037481766513715358449279314556797352446246 \\
86031794675012447568947440627494235603593650167405093344890920102983452226732247771897083232172 \\
820515736450136036130423677827163618778179383743982431301907362478638761841403754168112028404465 \\
938319290746385252639208772430477545163127182525068111451400502733404381769675255289127346639350 \\
0982215708440038078851633249658388252243638191808002001670321863502451077513469795963146961536667 \\
16168951481948091060066730184766785177739430387542983086720546391814425684391173074726514615419 \\
3438041627833661739775057161236346096236566875251277843062329973044475486561062204356908568471471 \\
27938378103853881888446379698900607607984328172725020839705886436071213605751867074569485840723 \\
7891694292536914086841719647957348103271148102172916286597358817409638991330560767785803399636173 \\
490553715036202472051577260781208855505433105576570014211875602940633575763850457503079087074 \\
37658530447052041130242629225537571145757355286060236699317039454793267182811289614232751427875 \\
69425690532832833440496355213025960089719251203669529880729403296453095969137708720454634896013 \\
276009554410598019825525493202412831593891984788152417957691939817112366182063687529915365150361 \\
1802144512343876568832561493559944050511495895691630753070266479560356836715895464485399551327261 \\
1203493865596129185620342224768038702907847352095116033447252547507168067262366158729727032960618 \\
2512044312194357156139201340952037872975243254476801554937002122953415949407262137232099852298394 \\
8384229076431913976732902383441830460409758599159285365304456971453176680449737096483324156185041
\end{align*}
Complexity considerations

The quadratic systems we obtain using $X^{q+1} + BX + B$ are not bilinear $\implies$ we can’t argue for the same $L_Q(1/4 + o(1))$ complexity that arises when using $X^q - X$. 
The quadratic systems we obtain using \( X^{q+1} + BX + B \) are not bilinear \( \implies \) we can’t argue for the same \( L_Q(1/4 + o(1)) \) complexity that arises when using \( X^q - X \).

However, when using \( X^q - X \), with judiciously chosen parameters, the complexity can be improved.
Complexity considerations

The quadratic systems we obtain using $X^{q+1} + BX + B$ are not bilinear $\implies$ we can’t argue for the same $L_Q(1/4 + o(1))$ complexity that arises when using $X^q - X$.

However, when using $X^q - X$, with judiciously chosen parameters, the complexity can be improved.

For $\mathbb{F}_{(q^k)^n}$ with $k \geq 2$ fixed, $n \approx q$ and $q \to \infty$ we showed that the DLP can be solved in time

$$L_{q^kn}(1/4, (\omega/8)^{1/4})$$
The algorithm of Barbulescu, Gaudry, Joux and Thomé

[BGJT13] have proposed a quasi-polynomial algorithm for the DLP in finite fields of small characteristic (eprint.iacr.org/2013/400).
The algorithm of Barbulescu, Gaudry, Joux and Thomé

[BGJT13] have proposed a quasi-polynomial algorithm for the DLP in finite fields of small characteristic (eprint.iacr.org/2013/400).

- Our relation generation gives an analogous quasi-polynomial algorithm; to eliminate $Q(x)$ just substitute $x$ by $Q(x)$.
The algorithm of Barbulescu, Gaudry, Joux and Thomé

[BGJT13] have proposed a quasi-polynomial algorithm for the DLP in finite fields of small characteristic (eprint.iacr.org/2013/400).

- Our relation generation gives an analogous quasi-polynomial algorithm; to eliminate $Q(x)$ just substitute $x$ by $Q(x)$
- In fact, our relation generation method and Joux’s, based on Möbius transforms of $X^q - X$, are equivalent
The algorithm of Barbulescu, Gaudry, Joux and Thomé

[BGJT13] have proposed a quasi-polynomial algorithm for the DLP in finite fields of small characteristic (eprint.iacr.org/2013/400).

- Our relation generation gives an analogous quasi-polynomial algorithm; to eliminate $Q(x)$ just substitute $x$ by $Q(x)$
- In fact, our relation generation method and Joux’s, based on Möbius transforms of $X^q - X$, are equivalent

For the BGJT algorithm, one setup issue is to find a set of coset representatives for $PGL_2(\mathbb{F}_{q^k})/PGL_2(\mathbb{F}_q)$:
The algorithm of Barbulescu, Gaudry, Joux and Thomé

[BGJT13] have proposed a quasi-polynomial algorithm for the DLP in finite fields of small characteristic (eprint.iacr.org/2013/400).

- Our relation generation gives an analogous quasi-polynomial algorithm; to eliminate $Q(x)$ just substitute $x$ by $Q(x)$
- In fact, our relation generation method and Joux’s, based on Möbius transforms of $X^q - X$, are equivalent

For the BGJT algorithm, one setup issue is to find a set of coset representatives for $PGL_2(\mathbb{F}_{q^k})/PGL_2(\mathbb{F}_q)$:

- $|PGL_2(\mathbb{F}_{q^k})/PGL_2(\mathbb{F}_q)| = (q^{3k} - q^k)/(q^3 - q) \approx q^{3k-3}$
The algorithm of Barbulescu, Gaudry, Joux and Thomé

[BGJT13] have proposed a quasi-polynomial algorithm for the DLP in finite fields of small characteristic (eprint.iacr.org/2013/400).

- Our relation generation gives an analogous quasi-polynomial algorithm; to eliminate \(Q(x)\) just substitute \(x\) by \(Q(x)\).
- In fact, our relation generation method and Joux’s, based on Möbius transforms of \(X^q - X\), are equivalent.

For the BGJT algorithm, one setup issue is to find a set of coset representatives for \(\text{PGL}_2(\mathbb{F}_{q^k})/\text{PGL}_2(\mathbb{F}_q)\):

- \(|\text{PGL}_2(\mathbb{F}_{q^k})/\text{PGL}_2(\mathbb{F}_q)| = (q^{3k} - q^k)/(q^3 - q) \approx q^{3k-3}\)
- For \(k = 2\), our search space has cardinality \(q^3 - q^2 \approx q^{3k-3}\)
The algorithm of Barbulescu, Gaudry, Joux and Thomé

[BGJT13] have proposed a quasi-polynomial algorithm for the DLP in finite fields of small characteristic (eprint.iacr.org/2013/400).

- Our relation generation gives an analogous quasi-polynomial algorithm; to eliminate $Q(x)$ just substitute $x$ by $Q(x)$
- In fact, our relation generation method and Joux’s, based on Möbius transforms of $X^q - X$, are equivalent

For the BGJT algorithm, one setup issue is to find a set of coset representatives for $PGL_2(\mathbb{F}_{q^k})/PGL_2(\mathbb{F}_q)$:

- $|PGL_2(\mathbb{F}_{q^k})/PGL_2(\mathbb{F}_q)| = (q^{3k} - q^k)/(q^3 - q) \approx q^{3k-3}$
- For $k = 2$, our search space has cardinality $q^3 - q^2 \approx q^{3k-3}$
- For $k \geq 3$ our search space has cardinality

$$q^k(q^k - 1)(q^k - \{q, q^2\})/(q^3 - q) \approx q^{3k-3}$$
A quasi-polynomial lower bound for Index Calculus?

For small characteristic fields of bitlength $l$, the BGJT algorithm has quasi-polynomial complexity $l^{O(\log l)}$.

- Applies to fields of the form $\mathbb{F}_{q^{kn}}$, with $k \geq 2$ and $n \approx q$
- Complexity dictated by $\#$ nodes in the descent tree
A quasi-polynomial lower bound for Index Calculus?

For small characteristic fields of bitlength $l$, the BGJT algorithm has quasi-polynomial complexity $l^{O(\log l)}$.

- Applies to fields of the form $\mathbb{F}_{q^{kn}}$, with $k \geq 2$ and $n \approx q$
- Complexity dictated by $\#$ nodes in the descent tree

**Question:** Are there any elements of $\mathbb{F}_{q^{kn}}$ that require a quasi-polynomial number of linear elements to represent them?
A quasi-polynomial lower bound for Index Calculus?

For small characteristic fields of bitlength $l$, the BGJT algorithm has quasi-polynomial complexity $l^{O(\log l)}$.

- Applies to fields of the form $\mathbb{F}_{q^{kn}}$, with $k \geq 2$ and $n \approx q$
- Complexity dictated by \# nodes in the descent tree

**Question:** Are there any elements of $\mathbb{F}_{q^{kn}}$ that require a quasi-polynomial number of linear elements to represent them?

**Answer:** No!
A quasi-polynomial lower bound for Index Calculus?

For small characteristic fields of bitlength \( l \), the BGJT algorithm has quasi-polynomial complexity \( l^{O(\log l)} \).

- Applies to fields of the form \( \mathbb{F}_{q^{kn}} \), with \( k \geq 2 \) and \( n \approx q \)
- Complexity dictated by \# nodes in the descent tree

**Question:** Are there any elements of \( \mathbb{F}_{q^{kn}} \) that require a quasi-polynomial number of linear elements to represent them?

**Answer:** No! F.R.K. Chung has proven that if \( \mathbb{F}_{q^{kn}} = \mathbb{F}_{q^k}(x) \), then each \( h \in \mathbb{F}_{q^{kn}}^\times \) can be represented by

\[
h = (x + a_1) \cdots (x + a_m), \quad \text{with} \quad a_i \in \mathbb{F}_{q^k},
\]

if \( \sqrt{q^k} > n - 1 \) and \( m \geq 2n + 4n \log n/(\log q^k - 2 \log (n - 1)) \).
Concrete security of small characteristic pairings

Adj, Menezes, Oliveira and Rodríguez-Henríquez recently studied the security of pairing fields once thought to be 128-bit secure.
Concrete security of small characteristic pairings

Adj, Menezes, Oliveira and Rodríguez-Henríquez recently studied the security of pairing fields once thought to be 128-bit secure.

In particular they showed that:

- The DLP in the 804-bit order $r$ subgroup of $\mathbb{F}_{36^{509}}^\times$ can be solved in time $2^{73.7} M_r$, using $q = 3^6$ and $k = 2$

- The DLP in the 698-bit order $r$ subgroup of $\mathbb{F}_{212^{367}}^\times$ can be solved in time $2^{94.6} M_r$, using $q = 2^{12}$ and $k = 2$
Concrete security of small characteristic pairings

Adj, Menezes, Oliveira and Rodríguez-Henríquez recently studied the security of pairing fields once thought to be 128-bit secure.

In particular they showed that:

- The DLP in the 804-bit order $r$ subgroup of $\mathbb{F}_{36 \cdot 509}^{\times}$ can be solved in time $2^{73.7} M_r$, using $q = 3^6$ and $k = 2$
- The DLP in the 698-bit order $r$ subgroup of $\mathbb{F}_{212 \cdot 367}^{\times}$ can be solved in time $2^{94.6} M_r$, using $q = 2^{12}$ and $k = 2$

Relies on existence of $f(X)$ of degree $n$ and $h_0(X), h_1(X)$ of small degree such that

$$h_1(X)X^q - h_0(X) \equiv 0 \pmod{f(X)}$$
New work [G. and Zumbrägel]: basic insight: use $f(X)$ of degree $n$ and $h_0(X), h_1(X)$ of small degree such that

$$h_1(X^q)X - h_0(X^q) \equiv 0 \pmod{f(X)}$$
New work [G. and Zumbrägel]: basic insight: use \( f(X) \) of degree \( n \) and \( h_0(X), h_1(X) \) of small degree such that

\[
h_1(X^q)X - h_0(X^q) \equiv 0 \pmod{f(X)}
\]

**Table:** Upper bounds on DLP security of the fields \( \mathbb{F}_{2^{12p}} \)

<table>
<thead>
<tr>
<th>( p )</th>
<th>79</th>
<th>103</th>
<th>127</th>
<th>199</th>
<th>239</th>
<th>313</th>
<th>367</th>
<th>439</th>
</tr>
</thead>
<tbody>
<tr>
<td>( # M_r )</td>
<td>( 2^{54} )</td>
<td>( 2^{54} )</td>
<td>( 2^{54} )</td>
<td>( 2^{54} )</td>
<td>( 2^{54} )</td>
<td>( 2^{76} )</td>
<td>( 2^{76} )</td>
<td>( 2^{76} )</td>
</tr>
</tbody>
</table>

\( \{79, \ldots, 239\} \) use \( q = 2^6 \) and \( k = 4 \); rest use \( q = 2^8 \) and \( k = 3 \).
New work [G. and Zumbrägel]: **basic insight**: use \( f(X) \) of degree \( n \) and \( h_0(X), h_1(X) \) of small degree such that

\[
h_1(X^q)X - h_0(X^q) \equiv 0 \pmod{f(X)}
\]

**Table**: Upper bounds on DLP security of the fields \( \mathbb{F}_{2^{12p}} \)

<table>
<thead>
<tr>
<th>( p )</th>
<th>79</th>
<th>103</th>
<th>127</th>
<th>199</th>
<th>239</th>
<th>313</th>
<th>367</th>
<th>439</th>
</tr>
</thead>
<tbody>
<tr>
<td>( #M_r )</td>
<td>( 2^{54} )</td>
<td>( 2^{54} )</td>
<td>( 2^{54} )</td>
<td>( 2^{54} )</td>
<td>( 2^{54} )</td>
<td>( 2^{76} )</td>
<td>( 2^{76} )</td>
<td>( 2^{76} )</td>
</tr>
</tbody>
</table>

\{79, \ldots, 239\} use \( q = 2^6 \) and \( k = 4 \); rest use \( q = 2^8 \) and \( k = 3 \).

- DLP in the 1221-bit order \( r \) subgroup of \( \mathbb{F}_{2^{24 \cdot 1223}}^{\times} \) can be solved in \( \approx 2^{95} M_r \) (we’re still optimising), using \( q = 2^{10} \) and \( k = 2 \).
Thanks for your attention!

Questions?