The Frobenius expansion discrete logarithm problem

Steven Galbraith

Royal Holloway, University of London

http://www.isg.rhul.ac.uk/~sdg/
This talk gives a brief survey of applications of Frobenius maps in elliptic curve cryptography. I will also sketch some recent research in collaboration with Phil Eagle, Waldyr Benits Jr. and Mike Scott. Special thanks are due to Tanja Lange.

- Introduction to Koblitz curves and Frobenius expansions and their applications.
- Compression of points on Koblitz curves.
- Using Frobenius expansions in cryptographic protocols.
- Frobenius expansion discrete logarithm problem.
- Efficient exponentiation in pairing-friendly groups.
Neal Koblitz suggested the elliptic curves

\[ E : y^2 + xy = x^3 + ax^2 + 1 \]

over \( \mathbb{F}_2 \) where \( a \in \{0, 1\} \).

If \( m \) is prime then there is a chance that \( E(\mathbb{F}_{2^m}) \) has almost prime group order and is suitable for elliptic curve cryptography.

Denote by \( r \) the large prime dividing \( \#E(\mathbb{F}_{2^m}) \).
Frobenius map

Exponentiation in $E(\mathbb{F}_{2^m})$ is sped-up by exploiting the Frobenius map

$$\tau(x, y) = (x^2, y^2).$$

Let $\mu = (-1)^{1+a}$ where $a$ is the coefficient of $x^2$ of the curve.

Then $\tau$ satisfies the polynomial

$$\tau^2 - \mu \tau + 2 = 0.$$ 

In other words $\tau(\tau(P)) - [\mu] \tau(P) + [2]P = \infty$ for all $P = (x_P, y_P) \in E(\mathbb{F}_{2^m})$. 
Koblitz noted that an integer $n$ may be expressed (non-uniquely) as a Frobenius expansion

$$n = \sum_{i=0}^{N} a_i \tau^i$$

where $a_i \in \{-1, 0, 1\}$. Since $\tau^m = 1$, we have $N < m$.

Hence, $[n]P$ can be computed very efficiently as

$$[n]P = \sum_{i=0}^{m} [a_i] \tau^i(P).$$
Remarks on Frobenius expansions

We use a normal basis for $\mathbb{F}_{2^m}/\mathbb{F}_2$, i.e., a vector space basis of the form

$$\{\theta, \theta^2, \theta^4, \ldots, \theta^{2^{m-1}}\}.$$

Note that $\theta^{2^m} = \theta$ is automatically satisfied.

Therefore, we can represent a field element as a binary string, and squaring is just a left shift with wrap-around.

One also uses a non-adjacent form (i.e., $a_i a_{i+1} = 0$) for extra efficiency.
Then the representation is unique if $N < m$. 
History of Frobenius expansions on Koblitz curves

- Koblitz (Crypto ’91) showed how to write $n$ as a Frobenius expansion.
- Meier-Staffelbach (Crypto ’92) made it shorter.
- Solinas (Crypto ’97, DCC 2000) made it shorter again.
- Generalisations to other curves were made by Müller and Günther-Lange-Stein etc.
It is not all good news

Wiener-Zuckerato and Gallant-Lambert-Vanstone showed how to speed-up Pollard methods using equivalence classes.

In particular, one can define the equivalence class

$$\{\pm\tau_i(P) : 0 \leq i < m\}$$

of size $2m$ for any point $P$ in $E(\mathbb{F}_{2^m})$ which does not lie in a subfield

Hence the expected number of group operations of the Pollard methods in $E(\mathbb{F}_{2^m})[r]$ is $\sqrt{2m}$ fewer than in a generic group of order $r$. 
This is joint work with Phil Eagle.

Due to the WZ/GLV equivalence classes, when using Koblitz curves one should work over a slightly larger field than when using a more general elliptic curve.

Precisely, to get $k$ bits of security with ECC it usually suffices to work over a $2k + 1$ bit finite field. But for Koblitz curves need to work over a $2k + 1 + \log_2(m)$ bit finite field. It follows that the bandwidth when using Koblitz curves is usually not optimal.
Sample sizes of $m$ for Koblitz curves are listed below. Here ‘Bytes to send’ is $\lceil (m + 1)/8 \rceil$ and ‘Security level’ is $(m - 1 - \log_2(m))/2)$. 

<table>
<thead>
<tr>
<th>$m$</th>
<th>Security level</th>
<th>$\log_2(m)$</th>
<th>Bytes to send</th>
<th>Desired num bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>163</td>
<td>77.3</td>
<td>7.3</td>
<td>21</td>
<td>20</td>
</tr>
<tr>
<td>233</td>
<td>112.1</td>
<td>7.8</td>
<td>30</td>
<td>28</td>
</tr>
<tr>
<td>283</td>
<td>136.9</td>
<td>8.1</td>
<td>36</td>
<td>35</td>
</tr>
</tbody>
</table>
Given a point \( P = (x_P, y_P) \) to be compressed do:

- Forget \( y_P \).
- Find the longest run of ones in the binary string \( x_P \), including runs which ‘wrap around’ the ends of the string.
  Let \( t \) be the length of the run.
- Perform bit shifts so that the binary string is of the form

\[
x_{m-1}x_{m-1} \cdots x_{t+2}011 \cdots 10
\]

Let \( \overline{x} = x_{m-1} \cdots x_{t+2} \) be the remaining binary string of length \( m - t - 2 \).

If the solution is not unique, choose the one so that \( \overline{x} \) is minimal.

Note that the all 0 and all 1 strings correspond to \( x_P = 0, 1 \). These will not arise in the applications.
Topic 1: Compressing points on Koblitz curves

Two options depending on the communication model:

- **Byte model:** If sending bytes then our only interest is to ensure that we send fewer bytes than using the naive method. Hence we perform the shifting as above and truncate a fixed number of bytes.

- **Bit model:** In principle one can remove the full $t + 2$ bits if one has a communication model which allows variable length binary strings to be sent.

- It is trivial to decompress.

- In general, one expects $t \approx \log_2(m)$. In practice one can save a byte of communication in the three examples earlier at least 90% of the time.
Topic 1: Compressing points on Koblitz curves

- How can this be used in cryptographic protocols?
- One must ensure that the protocol makes sense on equivalence classes.
- For example, with Diffie-Hellman, Alice computes $A = [a]P$ and then sends the compressed $x_A$.
  
  Bob computes $B = [b]P$ and sends $x_B$.

- Alice decompresses $x_B$ to get a point $B'$ which is equivalent, but usually not equal, to $B$.
  
  She then computes $K = [a]B'$ and finally $x_K$.

- Bob will similarly compute $K' = [b]A'$.
  
  One can show that $x_K = x_K'$.
Fast exponentiation using Frobenius expansions

As mentioned, for fast exponentiation we convert an integer $n$ into a Frobenius expansion (for example, using the Solinas algorithm)

$$n = \sum_{i=0}^{N} a_i \tau^i$$

where $a_i \in \{-1, 0, 1\}$.

The conversion algorithm is very simple and requires only integer addition and reduction modulo 4. Nevertheless, the conversion requires extra code. This could be a problem when using very constrained devices.

Why choose a random integer and then convert to a Frobenius expansion? Why not just start with a random Frobenius expansion?
Topic 2: Replacing integers with Frobenius expansions in cryptographic protocols

Joint work with Waldyr Benits Jr.

There are several issues to consider:

▶ Is the protocol still secure?
▶ If so, can you prove it?
▶ Is this actually more efficient?
  Since might need to convert Frobenius expansion back to an integer for further computations in the protocol (e.g., signature schemes).
Case study: The GPS identification scheme

(Girault, Poupard, Stern)
This is a protocol designed for constrained environments, where the online computations by the Prover must be as efficient as possible.

- Prover has public key $Q = [a]P$ and private key $a$.
- Prover pre-computes commitment $R = [r]P$ offline.
- In the online stage, the Prover sends $R$ and receives a short challenge $c$ from the Verifier.
- The Prover computes $s = r + ca$ and sends it to the Verifier.
- The verifier checks $[s]P = R + [c]Q$.
- Note that there is no modular reduction in the online stage.
- See Girault-Poupard-Stern J. Crypt for details of the parameter sizes.
It is natural to use Koblitz curves and Frobenius expansions to speed up the offline and verification computations in the GPS protocol.

The Prover then needs code for:

- elliptic curve operations using Frobenius expansions;
- generating random Frobenius expansions and conversion Frobenius expansions to integers;
- large integer arithmetic (to compute \( r + ca \)).
We adapted this protocol to use Frobenius expansions throughout. Precisely: $a$, $r$, $c$ and $s$ are all Frobenius expansions.

We give an algorithm to compute $r + ca$ so that the result is of the form

$$\sum_{i=0}^{N} a_i \tau^i$$

where $a_i \in \{-1, 0, 1\}$.

The Prover only needs code for elliptic curve arithmetic using Frobenius expansions, and arithmetic with Frobenius expansions. (Paper to appear in Weworc 2007.)
Conclusion: This is not a good idea.

- There is an attack. Hence one needs to perform a re-randomisation process when computing $r + ca$.
- The security proof seems to be much harder.
  - We only prove computational zero knowledge rather than statistical zero knowledge.
  - The parameters need to be larger for the security proof.
- The algorithm for computing $r + ca$ using Frobenius expansions is more complicated than the algorithms for converting between Frobenius expansions and integers.
Topic 3: The Frobenius expansion discrete logarithm problem

Joint work with Waldyr Benits Jr. Special thanks to Tanja Lange.

If one directly uses Frobenius expansions in cryptographic protocols a natural question is: How long do they have to be?

There are two issues to consider.

- Ensuring uniform distribution of group elements (see Lange-Shparlinski).
  The answer is ‘a little bit longer than you would like’. However, as we did for the GPS protocol, one might be able to prove security directly without needing uniform distribution properties.
- Ensuring that the DLP is hard.
The Frobenius expansion DLP

Given \( P, Q \in E(\mathbb{F}_{2^m}) \) and \( N \), find a Frobenius expansion

\[
a = \sum_{i=0}^{N} a_i \tau^i
\]

with \( a_i \in \{-1, 0, 1\} \) (if it exists) such that \( Q = [a]P \).

The number of inequivalent Frobenius expansions of length \( N \) is \( O(2^N) \).

Of course, one can just solve the standard ECDLP using Pollard methods and then convert the result to a Frobenius expansion. If \( N \approx m \) then this is the best thing to do. So the problem is mainly of interest when \( N \ll m \).
The Frobenius expansion DLP


She mentions that the standard Pollard methods cannot trivially be used to solve this problem.

She states “One can design a $\tau$-adic baby-step-giant-step algorithm” but gives no details.
Frobenius expansions

Frobenius expansions are elements of the ring $\mathbb{Z}[\tau]/(\tau^2 - \mu \tau + 2)$. Hence

$$\sum_{i=0}^{N} a_i \tau^i \equiv b_0 + b_1 \tau$$

for some $b_i \in \mathbb{Z}$.

Indeed, we have

$$|b_0| < c_0 \sqrt{2^N}, \quad |b_1| < c_1 \sqrt{2^N}$$

for some constants $c_0$ and $c_1$.

I conjecture $(c_0, c_1) = (3, 2)$ but can only prove $(c_0, c_1) = (3.7, 2.6)$. 
Restatement of Frobenius expansion DLP

Given \( P, Q \in E(\mathbb{F}_{2^m}) \) and \( N \), find a Frobenius expansion

\[
a = b_0 + b_1 \tau
\]

with \( |b_0| < c_0 \sqrt{2^N} \), \( |b_1| < c_1 \sqrt{2^N} \) (if it exists) such that

\[
Q = [b_0]P + [b_1]\tau(P).
\]

The baby-step-giant-step algorithm is immediate.
Low memory algorithm

Gaudry and Schost proposed a 2-dimensional Pollard kangaroo method, and their idea can be used for this application.

Let \( M_1 = \lfloor c_1 \sqrt{2^N} \rfloor \) and \( M_2 = \lfloor c_2 \sqrt{2^N} \rfloor \).

- Partition \( E(\mathbb{F}_{2^m}) \) into \( k \) sets by defining a function

\[
  f : E(\mathbb{F}_{2^m}) \to \{1, \ldots, k\}.
\]

- Choose \( k \) random integers \( \beta_i \) in the range \(-M_2 \leq \beta_i \leq M_2\) such that \( \sum_{i=1}^{k} \beta_i = 0 \).

- Define the random walk

\[
  P_{n+1} = P_n + [\beta_f(P_n)] P + \tau(P).
\]
Low memory algorithm

- Start the tame Kangaroo at $P_1 = [-M] \tau(P)$.
- Iterate the random walk for $c_3 \sqrt{2^N}$ steps.
- Store the point $P_n$ and the coefficients $b_0, b_1$ of the representation
  \[ P_n = [b_0]P + [b_1] \tau(P). \]
- Start the wild kangaroo at $Q_1 = Q$.
- Iterate the random walk and check at each stage whether $Q_{n'} = P_n$.
- A match gives
  \[ P_n = [b_0]P + [b_1] \tau(P) = Q_n = Q + [b'_0]P + [b'_1] \tau(P) \]
  and the Frobenius expansion DLP is solved.
Joint work with Mike Scott.

We construct a group homomorphism in certain pairing-friendly groups for which one can apply the Gallant-Lambert-Vanstone (GLV) technique for fast exponentiation.
Let $\psi$ be a homomorphism such that $\psi(P) = [\lambda]P$ for some $\lambda \in (\mathbb{Z}/r\mathbb{Z})$.

Let $n$ be an exponent.
Write $n \equiv n_0 + n_1 \lambda \pmod{r}$ where $|n_0|, |n_1| \leq \sqrt{r}$.

Then one can compute $[n]P$ as

$$[n_0]P + [n_1]\psi(P).$$

This multi-exponentiation is faster than computing $[n]P$. 
Pairing friendly groups

Let

\[ e : G_1 \times G_2 \to G_T \]

be a pairing where

\[ \begin{align*}
G_1 &= E(\mathbb{F}_p)[r], \\
G_T &\subset \mathbb{F}^*_p, \\
G_2 &= E'(\mathbb{F}_{p^e})[r] \text{ where } E' \text{ is a twist of } E \text{ and } e \mid k.
\end{align*} \]

We aim to speed up operations in \( G_2 \).
Let $Q \in G_2$ be given.
There is an isomorphism

$$\phi : E'(\mathbb{F}_{p^e}) \rightarrow E(\mathbb{F}_{p^k}).$$

One can prove that

$$\phi^{-1} \tau \phi : E'(\mathbb{F}_{p^e}) \rightarrow E'(\mathbb{F}_{p^e})$$

is a group homomorphism with eigenvalue $\lambda$. 
The homomorphism

When $e = 1$ one can prove that

$$\psi = \rho \tau'$$

where $\tau'$ is the $p$-power Frobenius on $E'$ and where $\rho$ is an element of $\text{Aut}(E')$.

Hence the result is interesting only when $e > 1$.

Feature: The decomposition of $n \equiv n_0 + n_1 \lambda \pmod{r}$ is often very simple, since $\lambda \approx \sqrt{r}$. 
Example: BN curves

The nicest family of pairing-friendly groups is by Barreto and Naehrig.

\[ t = 6x^2 + 1, \quad p = 36x^4 + 36x^3 + 24x^2 + 6x + 1, \quad r = p + 1 - t. \]

The embedding degree is \( k = 12 \).

The curve \( E' \) is defined over \( \mathbb{F}_{p^2} \), so \( e = 2 \).

The eigenvalue of \( \psi \) on \( E'(\mathbb{F}_{p^2}) \) is \( T = 6x^2 \).

Our homomorphism can be used to obtain a 4-dimensional GLV method.
Conclusion

We have given several applications of the Frobenius map in elliptic curve cryptography.

- Compressing points on Koblitz curves.
- Bad version of the GPS identification scheme.
- Study of the Frobenius expansion discrete logarithm problem.
- Speeding up exponentiation in the pairing group $G_2$.
- Some of these results are trivial to generalise to other fields or hyperelliptic curves.
Thank you