Computing Cryptographic Pairings: the State of the Art

Craig Costello
craig.costello@qut.edu.au
QUT and UCI

Winter 2010
University of California, Irvine
Pairing computation speeds: then and now

Then:
- 1993 Menezes’ elliptic curve book (post MOV attack): few minutes

...BIG GAP...

Now:
- 2009 Hankerson, Menezes, Scott: 4.01ms
- April 2010 Naehrig, Niederhagen, Schwabe: 1.80ms
- June 2010 Beuchat et al.: 0.94ms
- October 2010 Aranha et al.: 0.65ms
So what happened in the big gap?

- Heaps of exciting protocol stuff has happened...
  *ID-based encryption (IBE), ID-based key agreement, short signatures, group signatures, ring signatures, certificateless encryption, hierarchical encryption, predicate-based encryption, attribute-based encryption, .... and many many more!!!*

- Heaps of cool pairing optimizations have ‘followed’...
  - *Tate pairing instead of Weil pairing*
  - *denominator elimination*
  - *group choices and twisted curves*
  - *endomorphism rings and loop shortening*
  - *low rho-valued curves*
  - *pairing and towering-friendly fields*
  - *quick explicit formulas*
  - *... and many more!!!*
A mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$:

- $P \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$ and $e(P, Q) \in \mathbb{G}_T$: (cyclic) groups are all of prime order $r$ (usually)

- **Bilinear**: $e(aP, bQ) = e(P, Q)^{ab} = e(bP, aQ)$

- **Note**: $\mathbb{G}_1$ and $\mathbb{G}_2$ must be linearly independent

- $e(P, Q) = f(x_P, y_P, x_Q, y_Q) \in \mathbb{F}_{q^k}$
Groups involved: the \( r \)-torsion and Frobenius eigenspaces

- The points \( P \) and \( Q \) in the pairing come from the \( r \)-torsion \( E(\mathbb{F}_q)[r] = \mathbb{Z}_r \times \mathbb{Z}_r \).

\[ \mathbb{F}_q \text{ must be extended } (\mathbb{F}_{q^k}) \text{ to contain the entire } r \text{ torsion} \]

\( P \in G_1 = E(\mathbb{F}_q)[r] \quad \text{ and } \quad Q \in G_2 \subset E(\mathbb{F}_{q^k})[r] \)

- Frobenius endomorphism \( \pi_q(x, y) \mapsto (x^q, y^q) \)

\( G_1 = E[r] \cap \text{Ker}(\pi_q - [1]) \quad \text{ and } \quad G_2 = E[r] \cap \text{Ker}(\pi_q - [q]) \)

- Both eigenspaces are very (computationally) convenient.
The embedding degree $k$ and pairing-friendly curves

- $\#E(\mathbb{F}_q) = q + 1 - t \approx q$ and $\#E(\mathbb{F}_q) = hr$ \( (h \text{ small, } r \text{ big prime}) \)
- To contain entire $r$-torsion (both $G_1$ and $G_2$), must extend $\mathbb{F}_q$ to $\mathbb{F}_{q^k}$
- $k \in \mathbb{N}$ is smallest s.t. \( r \mid q^k - 1 \)
- In general, $k \approx r$ (Balasubramanian and Koblitz)
- Let’s be modest: $q = 160$ bits, $r = 160$ bits $\rightarrow \mathbb{F}_{q^k} \approx \mathbb{F}_{2^{160}(2^{160})}$
- Need to find ‘pairing-friendly’ elliptic curves where $k$ is small enough $k < 50$
- Finding pairing-friendly curves is an art in itself...
Attacker can target either discrete log problem: $E(\mathbb{F}_q)$ or $\mathbb{F}_{q^k}$

We aim to balance their difficulty to optimize implementation

Define $\rho = \log q / \log r$ (closer to 1 the better)

<table>
<thead>
<tr>
<th>(AES) Security level (bits)</th>
<th>Subgroup size $r$ (bits)</th>
<th>Extension field $q^k$ (bits)</th>
<th>Embedding degree $k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>160</td>
<td>960-1280</td>
<td>6-8 $\rho \approx 1$</td>
</tr>
<tr>
<td>112</td>
<td>224</td>
<td>2200-3600</td>
<td>10-16</td>
</tr>
<tr>
<td>128</td>
<td>256</td>
<td>3000-5000</td>
<td>12-20</td>
</tr>
<tr>
<td>192</td>
<td>384</td>
<td>8000-10000</td>
<td>20-26</td>
</tr>
<tr>
<td>256</td>
<td>512</td>
<td>14000-18000</td>
<td>28-36</td>
</tr>
</tbody>
</table>

Table: I stole this table from the “taxonomy” paper (Freeman, Scott, Teske)
Barreto and Naehrig found a family of really nice curves for \( k = 12 \)

\[
\begin{align*}
q(x) &= 36x^4 - 36x^3 + 24x^2 - 6x + 1 \\
\#E(\mathbb{F}_q)(x) &= 36x^4 - 36x^3 + 18x^2 - 6x + 1 \\
t(x) &= 6x^2 + 1
\end{align*}
\]

Find \( x \) s.t. \( q(x) \) is prime and \( \#E(x) \) is also prime and you have a BN curve \( y^2 = x^3 + b \)

In fact, almost all constructions (\( r \) prime) result in a curve \( y^2 = x^3 + b \) or \( y^2 = x^3 + ax \) (no CM needed)

The “bible”: Freeman-Scott-Teske - “A taxonomy of pairing-friendly elliptic curves”
The elements of $G_2$ are much bigger than the elements of $G_1$ (e.g. $k = 12$)

$$F_{q^{12}} = F_{q^4}(\alpha) = F_{q^2}(\gamma) = F_q(\beta)$$

$P \in G_1$: [341746248540, 710032105147]

$Q \in G_2$:

$$[((502478767360 \cdot \beta + 1034075074191) \cdot \gamma + 342970860051 \cdot \beta + 225764301423) \cdot \alpha^2 + ((205398279920 \cdot \beta + 182600014119) \cdot \gamma + 860891557473 \cdot \beta + 435210764901) \cdot \alpha + (1043922075477 \cdot \beta + 566889113793) \cdot \gamma + 150949917087 \cdot \beta + 21392569319, ((654337640030 \cdot \beta + 744622505639) \cdot \gamma + 1092264803801 \cdot \beta + 895826335783) \cdot \alpha^2 + ((529466169391 \cdot \beta + 550511036767) \cdot \gamma + 985244799144 \cdot \beta + 554170865706) \cdot \alpha + (194564971321 \cdot \beta + 969736450831) \cdot \gamma + (579122687888 \cdot \beta + 581111086076)]}$$
The twisted curve

- Original curve is $E(\mathbb{F}_q) : y^2 = x^3 + ax + b$
- Twisted curve is $E'((\mathbb{F}_q)^{k/d}) : y^2 = x^3 + a\omega^4 x + b\omega^6$, $\omega \in \mathbb{F}_q^k$
- Possible degrees of twists are $d \in \{2, 3, 4, 6\}$: the bigger the better!
- Twist $\Psi : E' \rightarrow E : (x', y') \rightarrow (x'/\omega^2, y'/\omega^3)$ induces $G'_2 = E'(\mathbb{F}_q^{k/d})[r]$ so that $\Psi : G'_2 \rightarrow G_2$
- Instead of working with $Q \in G_2$, a lot of work can be done with $Q' \in G'_2$ defined over subfield $\mathbb{F}_q^e = \mathbb{F}_q^{k/d}$

$P \in G_1 : (341746248540, 710032105147)$

$Q' \in G'_2 = \Psi^{-1}(G_2)$:

$(((917087150949\beta + 25693192139) \cdot \omega^2, (878885791226\beta + 860765811110) \cdot \omega^3)$
Achieving a bilinear pairing

- On elliptic curves, group homomorphism from points to divisor classes

\[ P \mapsto (P) - (\mathcal{O}) = D_P \]

- Let \( D \) be the divisor \( D = \sum_P n_P(P) \) on \( E \) and \( f \in \mathbb{F}_{q^k}(E) \):

\[ f(D) = \prod_P f(P)^{n_P} \]

- \( f, g \in \mathbb{F}_{q^k}(E) \): Weil reciprocity: \( f(\text{div}(g)) = g(\text{div}(f)) \)

- Achieve bilinearity (and other necessary properties) by finding a function \( f_P \) whose divisor is some (linear) multiple of \( D_P = (P) - (\mathcal{O}) \)...
Let $P \in E[r]$, (assume) we can construct the function $f_{v,P}$ such that
\[
\text{div}(f_{v,P}) = v(P) - ([v]P) - (v - 1)(O)
\]
When $v = r$, we have
\[
\text{div}(f_{r,P}) = r(P) - ([r]P) - (r - 1)(O)
\]
\[
= r(P) - r(O)
\]
\[
= rD_P
\]
\[
f_P = f_{r,P} \text{ is a degree } r \text{ function (has zero of degree } r \text{ at } P)\ldots
\]
\[
\text{Remember } r \text{ has to be large } > 2^{160} \text{ for ECDLP to be hard}
\]
### Weil vs. Tate pairings

#### Weil pairing

\[ e : G_1 \times G_2 \rightarrow \mu_r \in \mathbb{F}_{q^k}, \quad (P, Q) \mapsto f_{r,P}(Q)/f_{r,Q}(P) \]

#### Tate(-Lichtenbaum) pairing

\[ e : G_1 \times G_2 \rightarrow \mu_r \in \mathbb{F}_{q^k}, \quad (P, Q) \mapsto f_{r,P}(Q)^{q^k-1} \]

- **Weil pairing:** compute two degree \( r \) functions
- **Tate pairing:** compute one degree \( r \) function and exponentiate (much faster)
- Exponentiation is somewhat standard, so how to compute \( f_{r,P}(Q) \) efficiently
- 1986: Miller proposes efficient algorithm for \( f_{r,P}(Q) \) (“The Weil pairing, and it’s efficient calculation”)
Miller’s algorithm to compute $f_{r,P}(Q)$

$r = (r_{l-1}, \ldots, r_1, r_0)_2$
initialize: $U = P$, $f = 1$
for $i = l - 2$ to $0$ do
  a. i. Compute $f_{\text{DBL}}(U)$ in the doubling of $U$
      ii. $U \leftarrow [2]U$
      iii. $f \leftarrow f^2 \cdot f_{\text{DBL}}(U)(Q)$
  b. if $m_i = 1$ then
     i. Compute $f_{\text{ADD}}(U,P)$ in the addition of $U + P$
     ii. $U \leftarrow U + P$
     iii. $f \leftarrow f \cdot f_{\text{ADD}}(U,P)(S)$
Optimization: force $r(x)$ to have low Hamming-weight

$r = (r_{l-1}, \ldots, r_1, r_0)_2$

initialize: $U = P$, $f = 1$

for $i = l - 2$ to 0 do

a. i. Compute $f_{\text{DBL}}(U)$ in the doubling of $U$
   ii. $U \leftarrow [2]U$
   iii. $f \leftarrow f^2 \cdot f_{\text{DBL}}(U)(Q)$

b. if $m_i = 1$ then
   i. Compute $f_{\text{ADD}}(U, P)$ in the addition of $U + P$
   ii. $U \leftarrow U + P$
   iii. $f \leftarrow f \cdot f_{\text{ADD}}(U, P)(S)$
Optimization: avoid costly inversions and exploit exponentiation

\[ r = (r_{l-1}, \ldots, r_1, r_0) \]
\[ \text{initialize: } U = P, \ f = 1 \]
for \( i = l - 2 \) to 0 do
i. Compute \( f_{\text{DBL}}(U) \) in the doubling of \( U \)
ii. \( U \leftarrow [2]U \) \hspace{1cm} \///(DBL)
iii. \( f \leftarrow f^2 \cdot f_{\text{DBL}}(U)(Q) \)

- **Irrelevant factors:** Because the final value of \( f \) is exponentiated to \( (q^k - 1)/r \), any subfield factors accumulated in \( f \) can be ignored!

- **Projective coordinates:** Affine coordinates require inversions: use \((X:Y:Z)\) to represent \((x,y) = (X/Z, Y/Z)\) or some other projection
Optimization: lower degree Miller functions (loop shortening)

- Exploit the fact that since $Q \in \mathbb{G}_2 = E[r] \cap \text{Ker}(\pi_q - [q])$, a bilinear pairing with a much smaller degree (than $r$) if $Q$ is the first argument
- $e(Q, P) = f_{\lambda, Q}(P)^{(q^k-1)/r}$ where $\lambda \equiv q \mod r$
- Vercauteren ("Optimal pairings") and Hess ("Pairing lattices") prove that $\lambda$ can be achieved as small as $r^{1/\varphi(k)}$
- Most of the computations are performed on the first argument (now $Q \in E(\mathbb{F}_{q^k})$), but many less iterations required for the lower degree function
- Dubbed the (optimal) "ate" pairing (since it reverses the arguments of the "eta" pairing, and it is (generally) faster than the Tate pairing
Optimization: pairing and towering-friendly fields

- Koblitz-Menezes 2005: Build extension fields as towers of extensions (using irreducible binomials)
- e.g. $k = 24$ build $\mathbb{F}_{q^k}$ as

\[
\begin{align*}
N &= M[X]/(X^2 - Y) \\
M &= L[Y]/(Y^2 - Z) \\
L &= K[Z]/(Z^2 - T) \\
K' &= F_p[T]/(T'^2 - \beta) \\
K &= F_p[T]/(T^3 - \beta)
\end{align*}
\]

Fig. 1. Tower of pairing-friendly fields

- Arithmetic and implementation much easier $k = 2^i3^j$ means $m_k = 3^i5^j m_1$ (e.g. $m_{24} = 135 m_1$)
- Best way to tower: Benger-Scott WAIFI2010 paper
In the Tate pairing, point operations and line computations were performed on $P \in E(\mathbb{F}_q)$ (somewhat negligible compared to the dominant operations in $\mathbb{F}_{q^k}$ for larger $k$).

In the ate pairing, these operations are now performed in $\mathbb{F}_{q^{k/d}}$.

Important to optimize the combination of a point doubling $U \mapsto [2]U$ (resp. additions) and the line computations that contribute to $f_{\lambda, Q}$.
C-Hisil-Boyd-Gonzalez-Wong (Pairing09): fastest pairings for $y^2 = x^3 + c^2$ (special Weierstrass): homogenous projective coordinates achieve 8 subfield multiplications

C-Lange-Naehrig (PKC2010): “Faster pairings on curves with high-degree twists”:

i. $y^2 = x^3 + ax$ ($j = 1728$ or $D = 1$): weight-(1,2) coordinates achieve 10 subfield multiplications

ii. $y^2 = x^3 + b$ ($j = 0$ or $D = 3$): Projective coordinates achieve 9 subfield multiplications (used in recent record 0.65ms)
Other curve models

- Weierstrass curves are nice for pairings since the line computations are inherent in the point addition formulas.
- Edwards curves (also Jacobi-Quartics, Hessian etc) are far superior in standard ECC because of fast addition formulas.

![Diagram](image)

(a) $P_1 \neq P_2, P_1, P_2 \neq O', P_3 = P_1 + P_2$
(b) $P_1 = P_2 \neq O', P_3 = 2P_1$

Figure: Picture taken from Arene et al. Edward’s pairing paper

- Pairing-based cryptosystems need more than just pairings.
- Galbraith showed $E$ and $E'$ can’t both be written in Edwards form ("Edwards curves aren’t likely candidates for ate pairing which requires computations")...

Ate pairing on Edwards curves

- C-Lange-Naehrig (PKC2010): a bilinear pairing can be computed entirely on the twist $E'$
- Choose $E$ so that $E'$ can be written in Edwards form (it doesn’t matter that $E$ can’t)
- C-Lange-Naehrig: “The ate pairing on twisted Edwards curves” (work in progress)
Some recent results

i. Compute $f_{DBL}(U)$ in the doubling of $U$

ii. $U \leftarrow [2]U$

iii. $f \leftarrow f^2 \cdot f_{DBL}(U)(S)$

\[(DBL) \ [2] (x_1, y_1) = (x_3, y_3)\]

\[f_{DBL}(U)(x, y) = y - \lambda \cdot x - (y_1 - \lambda \cdot x_1)\]

\[f_{DBL}(U)(S) = y_S - \lambda \cdot x_S - (y_1 - \lambda \cdot x_1)\]

- Perhaps it isn’t optimal to evaluate indeterminate function $f_{DBL}(U)(x, y)$ yet
- Leave as an indeterminate function for $n$-iterations (CBGW - AfricaCrypt2010 paper, CBGW - WAIFI 2010 paper)
- Even more advantageous in the case of a fixed pairing argument (C-Stebila - “Fixed argument pairings” - LatinCrypt 2010)
\( e(R, S) \): \( R \)-dependent vs. \( S \)-dependent computations

a. i. Compute \( f_{\text{DBL}}(U) \) in the doubling of \( U \)
   
   ii. \( U \leftarrow [2]U \)  
   
   iii. \( f \leftarrow f^2 \cdot f_{\text{DBL}}(U)(S) \)

b. if \( m_i = 1 \) then
   
   i. Compute \( f_{\text{ADD}}(U,R) \) in the addition of \( U + R \)
   
   ii. \( U \leftarrow U + R \)  
   
   iii. \( f \leftarrow f \cdot f_{\text{ADD}}(U,R)(S) \)

- All the point operations and line coefficient computations are completely \( R \)-dependent (\( U = vR \) throughout)
- If \( R \) is a fixed argument, we can pre-compute all of this before we input (or know) \( S \)
- Pre-compute and store all the \((\lambda, x_{U_i}, y_{U_i})\) tuples (Scott 2006)
- **C-Stebila:** do much more with all of the \( f_{\text{ADD}} \) functions before \( S \) is known (or input)
Tate and ate $\mathbb{F}_p$-muls vs. storage cost ($k = 12$, $r = 256$)
Current/future work: genus 2 pairings

- Working in the Jacobian $\text{Jac}_C(\mathbb{F}_q)$
- The general belief is that genus 2 pairings won't be competitive with pairings on elliptic curves
- I'm naive in this arena and am therefore not yet convinced
- Holding genus 2 implementations back: $\rho$-values are currently very bad in comparison

$$\rho = g \log q / \log r$$

- At the top of my wish list: pairing-friendly genus 2 curves $k \leq 50$ and $\rho << 4$
Questions?