Bilinear Pairings in Cryptography: School Overview

Dan Boneh
Stanford University
In the beginning …

Crypto in $F_p^*$:

- Diffie-Hellman key exchange, pub-key encryption, digital signatures, …

But discrete log problem in $F_p^*$ is only sub-exp hard

$\text{GNFS: } \exp( \approx \log^{1/3}(p) )$, record = 530-bit prime
Discrete log in other finite groups?

Lots of other groups can be used for crypto:

- extension fields, matrix groups, class groups, ....

But either sub-exp Dlog or inefficient group operation

Elliptic curves over $\mathbb{F}_p$: [Miller’85, Koblitz]

- no known sub-exp Dlog algorithm, and
- efficient group operation

<table>
<thead>
<tr>
<th>Security Comparison</th>
<th>Symmetric</th>
<th>$\mathbb{F}_p^*$</th>
<th>ECC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>128 bits</td>
<td>3092 bits</td>
<td>256 bits</td>
</tr>
</tbody>
</table>
Elliptic Curve Crypto: Day 1

Cryptosystems in $\mathbb{F}_p^*$ generally translate to elliptic curves.

Wide deployment:

Elliptic curve Diffie-Hellman
Pairings: additional structure on elliptic curves

(Some) elliptic curves have surprising structure [Weil 1940]

Pairing $e: E(F_p) \times E(F_p) \rightarrow F_{p^\alpha}$

2 points on curve

finite field extension

$s.t. \quad \forall \ P \in E(F_q), \ a,b \in \mathbb{Z}: \quad e(aP, bP) = e(P,P)^{ab}$

and $e(P,Q)$ is efficiently computable [Miller’86]
More abstractly: bilinear groups

$G, G_T$: finite cyclic groups of prime order $q$.

- **Def**: A **pairing** $e: G \times G \rightarrow G_T$ is a map:
  - **Bilinear**: $e(g^a, g^b) = e(g, g)^{ab} \quad \forall a, b \in \mathbb{Z}, g \in G$
  - **Poly-time computable and non-degenerate**: $g$ generates $G \implies e(g, g)$ generates $G_T$

- **Current examples**: $G \subseteq E(F_p), \quad G_T \subseteq (F_{p\alpha})^*$

  ($\alpha = 1, 2, 3, 4, 6, 10, 12$)
Pairings-based crypto: days 2,3,4

Encryption schemes with new properties:

- Identity-based, attribute-based, functional,
- Broadcast, BGN-Homomorphic, Searchable, CCA,

Signature systems with new properties:

- Short, Aggregate, Append-only, VRF,
- Short group sigs, e-cash, anon. credentials

Protocols:

- 3-way DH, efficient NIZKs, SNARGs,
Simplest example: BLS signatures

KeyGen: \[ sk = \text{rand. } x \text{ in } \mathbb{Z}_q, \quad pk = g^x \in G \]

Sign\((sk, m) \rightarrow H(m)^x \in G\) \[ e(g, H(m)^x) = e(g^x, H(m)) \]

Verify\((pk, m, \text{sig}) \rightarrow \text{accept iff } e(g, \text{sig }) \overset{?}{=} e(pk, H(m)) \]

Thm: Existentially unforgeable under CDH in the RO model

New properties: (unknown with \(F_p^*\))

- **Short:** signature is a single element in \(G\)
- **Aggregatable** [BGLS’02]
Aggregating BLS signatures

Verifying an aggregate signature:

(incomplete)

Applications: cert. chains, bitcoins, SBGP

Verifying an aggregate signature: (incomplete)

$$\prod_{i=1}^{n} e(H(M_i), h^{x_i}) = e(S, h)$$

$$\prod_{i=1}^{n} e(H(M_i)^{x_i}, h) = e(\prod_{i=1}^{n} H(M_i)^{x_i}, h)$$
How pairings work: Day 4

Miller’s algorithm and optimizations

Basis of several pairings implementations:
- PBC, jPBC, TinyPBC, MIRACL
Beyond bilinear maps [BS’03, GGH’12]

k-linear map \( e : G \times G \times \cdots \times G \rightarrow G_k \)

- non-degenerate, efficient, hard Dlog in \( G \)

Even more applications:
- Optimal broadcast encryption,
- optimal traitor tracing,
- ABE for circuits [SW’12], ...

Can they be constructed?
k-linear maps: a recent breakthrough
S. Garg, C. Gentry, S. Halevi

**Properties: (informal)**

- “randomized” representation of group elements
- Representation of $g \in G$ is $O(k)$ bits
- More than $k$-linear map: **gradation**

\[
e_1 : G \times G \rightarrow G_2
\]
\[
e_2 : G \times G_2 \rightarrow G_3
\]
\[
\vdots
\]
\[
e_k : G \times G_k \rightarrow G_{k+1}
\]
The future

- Lots of work to do to extend and enhance bilinear constructions using k-linear maps

- Many (but not all) bilinear techniques translate to lattices
  - On going effort -- more on this tomorrow
  - Many open questions: 3-way DH, broadcast enc., …

… but first need to understand bilinear techniques