Introduction to Elliptic Curve Cryptography

Anupam Datta

18-733
Elliptic Curve Cryptography

• Public Key Cryptosystem

• Duality between Elliptic Curve Cryptography and Discrete Log Based Cryptography
  – Groups / Number Theory Basis
  – Additive group based on curves

• What is the point?
  – Less efficient attacks exist so we can use smaller keys than discrete log / RSA based cryptography
Computing Dlog in \((\mathbb{Z}_p)^*\) (n-bit prime \(p\))

Best known algorithm (GNFS):

\[
\text{run time } \quad \exp\left(\tilde{O}\left(\sqrt[3]{n}\right)\right)
\]

<table>
<thead>
<tr>
<th>cipher key size</th>
<th>modulus size</th>
<th>Elliptic Curve group size</th>
</tr>
</thead>
<tbody>
<tr>
<td>80 bits</td>
<td>1024 bits</td>
<td>160 bits</td>
</tr>
<tr>
<td>128 bits</td>
<td>3072 bits</td>
<td>256 bits</td>
</tr>
<tr>
<td>256 bits (AES)</td>
<td>15360 bits</td>
<td>512 bits</td>
</tr>
</tbody>
</table>

As a result: slow transition away from \((\text{mod } p)\) to elliptic curves
Discrete Logs

• Let $p = 2q + 1$ where $p, q$ are large primes
• $\mathbb{Z}_p$ is the group of integers modulo $p$
• $|\mathbb{Z}_p| = 2q$
• $G_q = QR(\mathbb{Z}_p)$ is the quadratic residue subgroup of $\mathbb{Z}_p$
• $|QR(\mathbb{Z}_p)| = q$, subgroup of prime order
• Every element $g \in G_q$ is a generator, pick a random one
• Pick secret $x$, compute $g^x \mod p$
• Public: $(p, q, g, g^x)$  Secret: $x$

• Discrete Log Assumption: Given Public it is hard to find Secret
Outline

• Elliptic curves over reals
• Elliptic curves over $\mathbb{Z}_p$
• ECDH and ECDSA
Consider the following equation:

\[ y^2 = x^3 + ax + b \]

Idea: we pick \((a, b)\) and form a group which is a set containing all of the points that satisfy the equation.

This group will be defined with a very special addition operation which introduces an additional imaginary point.
Example
Not all curves are valid elliptic curves

- Left: $y^2 = x^3$ has a “cusp”
- Right: $y^2 = x^3 - 3x + 2$ has a “self intersection”

- In general we require: $4a^3 + 27b^2 \neq 0$

- Observation: curves are symmetric about the point $y = 0$
Elliptic Curves as a Group

• Groups are sets defined over some operation with some structure / properties

• \( G = \{(x, y): y^2 = x^3 + ax + b\} \)

• Define an operation denoted by ‘+’ such that:
  • If \( p_1, p_2 \in G, \ p_1 + p_2 \in G \) (Closure)
  • \((p_1 + p_2) + p_3 = p_1 + (p_2 + p_3)\) (Associative)
  • \( \exists 0 \ s.t. \ \forall p \ p + 0 = 0 + p = p \) (Identity)
  • \( \forall p \ \exists p^{-1} \ s.t. \ p + p^{-1} = 0 \) (Inverse)

– Curves will form an abelian group
  • \( p_1 + p_2 = p_2 + p_1 \) (Communitive)
The Group Operation

• Not typical point-wise addition!

• What is this 0 element?
  \[ y^2 = x^3 + ax + b \] does not include \((0, 0)\) if \(b \neq 0\)

• How do we know inverses exist if we don’t know what the 0 element is?

• How do we maintain closure?
  \[ (x, y) + (x, y) = (2x, 2y) \] for typical pointwise addition
  which in general does not lie on the curve
The Group Operation

• Let $P, Q, R \in G$, such that a line passes through all of them, then group operation is:

$$P + Q + R = 0$$

• This is strange, we have a relationship between points that lie along but no clear notion of traditional addition

• We can use the relationship to define a more traditional form of addition:

$$P + Q = -R$$
The Group Operation

- \( P + Q = -R \)
- \( R = (x_r, y_r), \quad -R = (x_r, -y_r) \)
- What happens if we want to compute \(- R + R\)?
  - What third point on the curve lies on the line defined by \((R, -R)\)?

- We say this is the point defined at infinity, we denote it by 0, and it is the additive identity
- \(-R + R = 0\)

- Adjust our definition of the group:
- \( G = \{(x, y): y^2 = x^3 + ax + b\} \cup \{0\} \)
The Group Operation (Geometric)

• Given $G = \{(x, y): y^2 = x^3 + ax + b\} \cup \{0\}$, calculate $P + Q$
  – Geometrically, figure out the third point $R$ such that a line goes through $P, Q, R$ and then set $P + Q = -R$

• What could possibly go wrong?
  – $P$ or $Q$ could be 0
    • $0$ is the identity under the group operation, so $P + 0 = 0 + P = P$
  – $P = -Q$
    • This is the case of $-R + R = 0$ which was defined by the vertical line
  – $P = Q$
    • Imagine tangent to $P$, use that to find $R$. $P + P = -R$ describes the line tangent to $P$ that intersects at $R$
  – There is no 3rd point
    • This occurs when the line is tangent to exactly one of $P$ or $Q$. Suppose the line is tangent to $P$, then from before we have $P + P = -Q$ which gives us $P + Q = -P$
    • If line is tangent to $Q$, then $Q + Q = -P$ which would give us $P + Q = -Q$
Algebraic Solution

• Let $P \neq Q$, line defined by $P, Q$ has slope

$$m = \frac{y_P - y_Q}{x_P - x_Q}$$

• Intersection with point $R = (x_R, y_R)$:
  - $x_R = m^2 - x_P - x_Q$
  - $y_R = y_P + m(x_R - x_P) = y_Q + m(x_R - x_Q)$

• How would we check that this is correct?
  - Check if $(x_R, y_R) \in G$, if it is then correct with high probability
Multiplication

• We have defined addition, so now we can define multiplication

\[ n \times P = P + P + \ldots + P \ (n \ - \ times) \]

• Inefficient for multiplying by large numbers

• Use doubling algorithm, analogue of repeated squaring algorithm for exponentiation

• Calculate 19 (6 Additions):
  – A = 1+1 = 2
  – B = A + A = 2 + 2 = 4
  – C = B + B = 4 + 4 = 8
  – D = C + C = 16
  – 19 = D + A + 1
Back to Discrete Logs

• In the discrete log setting, exponentiation was easy, but logs were hard
  – $g^x$ — Easy, $\log_g g^x$ — Hard

• In the elliptic curve setting, multiplication is easy but division is hard
  – We still call division “logarithm” even though its really division here

• We used the asymmetry of these operations in the discrete log setting to do key exchange / encryption, can do a similar thing with elliptic curves
Fields

• A field is a set $\mathbb{F}$ with two operations $(+,	imes)$ that has the following properties:
  – $\mathbb{F}$ is an abelian group under $+$
  – The non-zero elements of $\mathbb{F}$ are an abelian group under $\times$
  – $a(b + c) = ab + ac \ \forall a, b, c \in \mathbb{F}$ (Distributive)
Elliptic Curves Over a Field

- Note: \( \mathbb{Z}^*_n(\,+\,\times) \) is a field when \( n \) is prime
- Refine the definition of the curve group again:
  \[
  G = \left\{ (x, y) \in (\mathbb{F}_p)^2 : y^2 = x^3 + ax + b \ (mod \ p) \right\} \cup \{0\}
  \land \ 4a^3 + 27b^2 \neq 0 \ (mod \ p)
  \]
- Curves are now defined only at discrete points and not over the smooth lines that we had before
Elliptic Curves Over a Field

\[ y^2 = x^3 - 7x + 10 \pmod{p} \] where \( p = 19, 97, 127, 487 \)
Operation for Curves Over a Field

Curve $y^2 = x^3 - x + 3 \ (mod \ 127)$, $P = (16, 20)$, $Q = (41, 120)$
Operation for Curves Over a Field

• The addition operation that we defined before works exactly the same on curves defined over a field

• All of the special cases are handled exactly the same as before

• Intersection with point $R = (x_R, y_R)$ still computed as:
  - $x_R = m^2 - x_P - x_Q \mod p$
  - $y_R = y_P + m(x_R - x_P) \mod p = y_Q + m(x_R - x_Q) \mod p$
Order of Elliptic Curve Group

• # of unique points in the group
  – Could simply try and count them, but there are too many for this to be possible

• Efficient algorithms for computing this exist
Subgroups of Elliptic Curve Groups

• In the discrete log setting, we selected a generator \( g \) and computed \( \{g^0, g^1, \ldots \} \mod p \)

• This group generated by the generator had an order that divided the order of the parent group by Lagrange’s Theorem

• In Elliptic Curves we can select a point \( P \) which is like a generator and compute \( \{0P, P, 2P, 3P, \ldots \} \mod p \), we call this a Base Point

• This operation will also generate a cyclic subgroup of the Elliptic curve group whose order divides the order of the parent group
Subgroups of Elliptic Curve Groups

• Suppose we pick a point, \( P \), how can we find the order of the subgroup generated by \( P \)?

• Let \( N \) be the order of the parent group

• Let \( N = p_1^{k_1} p_2^{k_2} \ldots \) be the prime factorization of \( N \)

• Let \( n \) be the order of the subgroup

• Idea: take all divisors of \( N \), given by the prime factorization, and sort them smallest to largest, call them \( n \). The order of the subgroup is the smallest \( n \) such that \( nP = 0 \).
Finding Base Point With High Order

• We will want to find a base point that generates a subgroup with prime order that is as high as possible

• Let \( h = \frac{N}{n} \) we will call \( h \) the **cofactor** of the subgroup

• Let \( n \) be the largest prime factor in the prime factorization of \( N \)

• \( NP = 0 \) because \( N \) is an integer multiple of any point \( P \)
• \( n(hP) = 0 \) by re-writing \( N = nh \)
• This tells us that the point \( hP = G \) has order \( n \) unless \( G = 0 \)
• \( G \) is a generator of a cyclic subgroup of prime order \( n \)
ECDH – Elliptic Curve Diffie-Hellman

• Regular Diffie-Hellman:
  – Alice has secret \( a \) and computes \( g^a \)
  – Bob has secret \( b \) and computes \( g^b \)
  – They exchange and compute \( g^{ab} \)
  – Key insight: it is hard for an adversary to compute \( g^{ab} \) from \( g^a, g^b \)

• ECDH Setting, Public Parameters: \((p, a, b, G, n, h)\)
  – \( p \) = large prime
  – \((a, b)\) = coefficients in \( y^2 = x^3 + ax + b \)
  – \( G \) = base point that generates subgroup of large prime order
  – \( n \) = order of the subgroup
  – \( h \) = cofactor of the subgroup
ECDH – Elliptic Curve Diffie-Hellman

- Alice: $d_A \leftarrow_R \mathbb{Z}_n, \quad H_A = d_A G$
- Bob: $d_B \leftarrow_R \mathbb{Z}_n, \quad H_B = d_B G$

- Alice -> Bob: $H_A$
- Bob -> Alice: $H_B$

- Alice: $d_A H_B = d_A d_B G$
- Bob: $d_B H_A = d_B d_A G$

- Say $S = d_A d_B G$ is the shared secret, can use it to derive a symmetric key
ECDSA – Elliptic Curve Digital Signature Algorithm

- Public Information: \((p, a, b, G, n, h)\)
- Alice’s Private Key: \(d_A\)
- Alice’s Public Key: \(H_A = d_A G\)

Alice signs a message \(m \in \mathbb{Z}_n\) by performing the following:

- \(k \leftarrow_R \mathbb{Z}_n\)
- \(P = kG = (x_P, y_P)\)
- \(r = x_P \mod n\), if \(r = 0\) start over
- \(s = k^{-1}(m + rd_A) \mod n\), if \(s = 0\) start over
- Output signature \((s, r)\)
ECDSA – Elliptic Curve Digital Signature Algorithm

- Bob can verify a message signed by performing the following:
  - Bob gets \( (m, s, r, H_A) \)
  - Calculate \( u_1 = s^{-1}m \mod n \), \( u_2 = s^{-1}r \mod n \)
  - Calculate \( P = u_1 G + u_2 H_A \)
  - Valid if and only if \( r = x_P \mod n \)
ECDSA – Elliptic Curve Digital Signature Algorithm

• Check that the algorithm is correct:
  – \( P = u_1 G + u_2 H_A = u_1 G + u_2 d_A G = (u_1 + u_2 d_A)G \)
  – \( P = (s^{-1}m + s^{-1}rd_A)G = s^{-1}(m + rd_A)G \)
  – \( s = k^{-1}(m + rd_A) \rightarrow k = s^{-1}(m + rd_A) \)

  – \( P = s^{-1}(m + rd_A)G = kG \) – Thus the signature will verify correctly
Acknowledgments

• Many slides created by Kyle Soska (TA for 18733 in Spring 2016)
Pairing Based Cryptography

- Computational Diffie-Hellman
  - Given \( g, g^a, g^b \) compute \( g^{ab} \)
- Decisional Diffie-Hellman
  - Given \( g, g^a, g^b \), can't tell \( g^{ab} \) apart from random element \( g^c \) for random \( c \)

Let \( G_1, G_2, G_T \) be groups of prime order \( q \), then a bilinear pairing denoted \( e \) is an operation that maps from \( G_1 \times G_2 \rightarrow G_T \) such that

- \( \forall a, b \in \mathbb{F}_q, \forall P \in G_1, \forall Q \in G_2 \; e(aP, bQ) = e(P, Q)^{ab} \neq 1 \)

Idea: We can use pairing based cryptography to create a situation where Computational Diffie-Hellman is hard, but Decisional Diffie-Hellman is easy.
Pairing Based Cryptography

• Computational Diffie-Hellman
  – Given $g, g^a, g^b$ compute $g^{ab}$

• Decisional Diffie-Hellman
  – Given $g, g^a, g^b$, cant tell $g^{ab}$ apart from random element $g^c$ for random $c$

• Suppose an adversary has $g^a, g^b, g^z$, where $g^z$ is randomly either $g^{ab}$ or $g^c$ for $c$ random. How can he check which one he has?
  – $e(g^a, g^b) = e(g, g)^{ab} = e(g, g^{ab})$
  – Adversary computes $e(g, g^z) =? e(g^a, g^b)$
Pairing Based Signatures (Boneh et al.)

• \( x \leftarrow_R \mathbb{Z}_q \)

• Private Key: \( x \), Public Key: \( g^x \)

• Sign message \( m \) by hashing it yielding \( h = H(m) \) and signing the hash as \( \sigma = h^x \)

• Verify \((\sigma, m)\) as \( e(\sigma, g) =? e(H(m), g^x) \)
  \[- e(\sigma, g) = e(h^x, g) = e(H(m)^x, g) = e(H(m), g)^x = e(H(m), g^x) \]
Twists Of Elliptic Curves

• Suppose you have an elliptic curve $E[p]$ over some field $\mathbb{F}$

• A twist of $E[p]$ another elliptic curve over a field extension of $\mathbb{F}$

• A twist of $E[p]$ will be isomorphic to $E[p]$, namely it will have the same order, and there is a 1-1 onto mapping between them
Other Notes

• Weil Pairing is a well studied pairing where the groups $G$ are elliptic curves

• There are many standardized elliptic curve groups
  - $y^2 + xy = x^3 + ax^2 + 1$ over $\mathbb{F}_{2^m}$, $m = \text{prime and } a = 0 \text{ or } 1$
    - Koblitz Curves, very fast addition and multiplication
  - $x^2 + y^2 = 1 + dx^2y^2$ where $d = 0 \text{ or } 1$
    - Edwards Curves, point addition is the same in all cases, and reasonably fast