Elliptic Curves

Elliptic curves are groups created by defining a binary operation (addition) on the points of the graph of certain polynomial equations in two variables. These groups have several properties that make them useful in cryptography. One can test equality and add pairs of points efficiently.

When the coefficients of the polynomial are integers, we can reduce the coefficients and points modulo a prime p to produce an interesting finite group whose order is near p. Choosing random coefficients results in groups with random orders near p.

One can use elliptic curves to factor integers, although probably not RSA moduli.

There is a probabilistic algorithm for proving primality that uses elliptic curves.

An elliptic curve group may be used directly in cryptographic algorithms in many of the same ways the multiplicative group of integers modulo p can be used. In these applications, the discrete logarithm problem is harder for elliptic curve groups than for the integers modulo p, permitting smaller parameters and faster algorithms.

To keep the formulas for the binary operation simple, we will restrict the polynomial to have the form $y^2 = x^3 + ax + b$, which is called the **Weierstrass form** of the elliptic curve.

Definition: An **elliptic curve** is the graph E or $E_{a,b}$ of an equation $y^2 = x^3 + ax + b$, where x, y, a and b are real numbers, rational numbers or integers modulo $m > 1$. The set E also contains a **point at infinity**, denoted ∞.

The point ∞ is not a point on the graph of $y^2 = x^3 + ax + b$. It will be the identity of the elliptic curve group.

The discriminant $b^2 - 4ac$ vanishes when the quadratic equation $ax^2 + bx + c = 0$ has a repeated root. For the cubic equation $x^3 + ax + b = 0$, the discriminant is $4a^3 + 27b^2$. It vanishes when the cubic has a repeated root. We will assume that that this discriminant is $\neq 0$, so that the cubic does not have a repeated root. Thus, we are excluding elliptic curves which have a "double point" or a "cusp."

If $P = (x, y)$ lies on the graph of $y^2 = x^3 + ax + b$, we define $-P = (x, -y)$, that is, $-P$ is P reflected in the x-axis.

Given two points P and Q, on the graph but not on the same vertical line, define $P + Q = -R$, where R is the third point on the straight line through P and Q.

If P and Q are distinct points on the graph and on the same vertical line, then they must have the form $(x, \pm y)$, that is, $Q = -P$, and we define $P + Q = \infty$, the identity element of the group.

Also, $P + \infty = \infty + P = P$ for any element P of the elliptic curve (including ∞).

To add a point $P \neq \infty$ to itself, draw the tangent line to the graph at P. If the tangent line is vertical, then $P = (x, 0)$ and we define $P + P = \infty$. If the tangent line is not vertical, then it intersects the graph in exactly one more point R, and we define $P + P = -R$. (If P is a point of inflection, then $R = P$.)

The addition rule may be expressed as

$$P + Q + R = \infty$$

if and only if P, Q, R are on the same straight line.

Theorem: An elliptic curve E with the addition operation $+$ forms an abelian group with identity ∞. The inverse of P is $-P$.

This means that the operation $+$ is well defined and assigns an element $P+Q$ of E to every pair of points P, Q of E. Also, $P+\infty = \infty+P = P$, $P+Q = Q+P$ and $(P+Q)+R = P+(Q+R)$ for all points P, Q, R of E.

Formula for the coordinates of $P + Q$

Let E be defined by $y^2 = x^3 + ax + b$. Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$.

If $x_1 = x_2$ and $y_1 = -y_2$, then $P = -Q$ and $P + Q = \infty$.

If $P \neq Q$, let s be the slope $s = (y_2 - y_1)/(x_2 - x_1)$ of the line through P and Q.

If $P = Q$, let s be the slope $s = (3x_1^2 + a)/(2y_1)$ of the tangent line to $y^2 = x^3 + ax + b$ at P. (Use implicit differentiation.)

Then $P + Q = (x_3, y_3)$, where $x_3 = s^2 - x_1 - x_2$ and $y_3 = s(x_1 - x_3) - y_1$.

Example: On the elliptic curve $y^2 = x^3 - 5x$, add the points $P = (-1, 2)$ and $Q = (0, 0)$.

Using the formula, we find that the slope is $s = (0 - 2)/(0 - (-1)) = -2$. Then $x_3 = (-2)^2 - (-1) - 0 = 5$ and $y_3 = (-2)(-1 - 5) - 2 = 10$, so $P + Q = (5, 10)$. One should check the arithmetic by verifying that the sum is a point on the curve. Here the check is $10^2 = 5^3 - 5 \cdot 5$.

Example: On the elliptic curve $y^2 = x^3 + 8$, compute $P + P$, where $P = (1, 3)$.

We use the second formula for the slope because $P = Q$. We have $s = (3 \cdot 1^2 + 0)/(2 \cdot 3) = 1/2$, $x_3 = (1/2)^2 - 1 - 1 = -7/4$ and $y_3 = (1/2)(1 - (-7/4)) - 3 = -13/8$, so $P + P = (-7/4, -13/8)$.

Note that if a and b and the coordinates of points P and Q on the elliptic curve $E_{a,b}$ are rational numbers, then the coordinates of $P+Q$ will be rational numbers (unless $P+Q=\infty$).

Therefore, if a and b and the coordinates of points P and Q on the elliptic curve $E_{a,b}$ are integers modulo m, then the coordinates of $P+Q$ will be integers modulo m, unless $P+Q=\infty$, provided that any division needed to add points is by a number relatively prime to m.

The modulus m cannot be even because we have to divide by 2 in the formula for the slope s when $P=Q$. The condition on the discriminant becomes $4a^3+27b^2 \not\equiv 0 \pmod{m}$.

Of course, the graph is not a curve in the plane; it is just a set of pairs of numbers modulo m.

Let us look at the points of the elliptic curve $y^2 \equiv x^3 + 3x + 4 \pmod 7$.

x	$(x^3 + 3x + 4) \bmod 7$	y
0	4	$2, 5$
1	1	$1, 6$
2	4	$2, 5$
3	5	none
4	3	none
5	4	$2, 5$
6	0	0

There are ten points on this elliptic curve, counting ∞.

Example: Add the points $(1,1)+(2,5)$ on the curve whose points were just listed.

We have $s = (5-1)/(2-1) = 4$, $x_3 = 4^2 - 1 - 2 = 13 \equiv 6 \pmod 7$ and $y_3 \equiv 4(1-6) - 1 \equiv 0 \pmod 7$, so the sum is $(6,0)$.

Example: Double the point $(2,2)$ on the same curve.

We must add $(2,2)+(2,2)$. We have $s = (3 \cdot 2^2 + 3)/(2 \cdot 2) \equiv 2 \pmod 7$, $x_3 = 2^2 - 2 \cdot 2 \equiv 0 \pmod 7$ and $y_3 \equiv 2(2-0) - 2 \equiv 2 \pmod 7$, so the sum is $(0,2)$.

Formula for the number of points on an elliptic curve modulo a prime.

Theorem: The number N of points on the elliptic curve $y^2 \equiv x^3 + ax + b \pmod{p}$ is $N = p + 1 + \sum_{x=0}^{p-1}((x^3 + ax + b)/p)$, where (r/p) is the Legendre symbol.

Proof: Each x between 0 and $p-1$ gives one value $x^3 + ax + b$. The number of y between 0 and $p-1$ with $y^2 \equiv x^3 + ax + b \pmod{p}$ is 0, 1, or 2 according as $x^3 + ax + b$ is a quadratic nonresidue, is $\equiv 0$, or is a quadratic residue, all modulo p. Counting ∞, we have

$$N = 1 + \sum_{x=0}^{p-1} \left(1 + \left(\frac{x^3 + ax + b}{p}\right)\right) =$$

$$= p + 1 + \sum_{x=0}^{p-1} \left(\frac{x^3 + ax + b}{p}\right).$$

Since there are as many quadratic residues as quadratic nonresidues in the interval $1 \leq r \leq p-1$, the Legendre symbol in the sum probably will be $+1$ about as often as it will be -1. Hence, we expect the number of points on a random elliptic curve modulo p to be close to $p+1$. H. Hasse proved that this is so.

Theorem: Let the elliptic curve E modulo a prime p have N points. Then

$$p + 1 - 2\sqrt{p} \leq N \leq p + 1 + 2\sqrt{p}.$$

When P is a point on an elliptic curve and k is a positive integer we write kP for the sum $P+P+\cdots+P$ of k P's. We also define $0P = \infty$ and $kP = (-k)(-P)$ when k is a negative integer. The fast exponentiation algorithm, with multiplication replaced by addition of points of an elliptic curve, provides a speedy way to compute kP. It takes $O(\log |k|)$ point additions to find kP when $k \neq 0$.

Let p be prime and let R_p denote an RSR modulo p. This set is a group with multiplication modulo p as the operation. Its size is $p-1$ elements.

Lenstra invented a factoring algorithm using elliptic curves. It is similar to Pollard's $p-1$ factoring algorithm, which computes a^L mod n for some large L. It finds a factor p of n when the order $p-1$ of the multiplicative group of integers modulo p divides L.

Lenstra's idea is to replace this group with an elliptic curve group $E_{a,b}$ modulo n. The algorithm begins at a random point P on this elliptic curve and computes LP for some large integer L, usually the product of all primes up to some limit. If the order of P in the elliptic curve modulo p divides L, then a gcd operation during one of the elliptic curve point additions will discover the p of n. The order of P is a random number in the Hasse interval

$$p + 1 - 2\sqrt{p} \leq N \leq p + 1 + 2\sqrt{p}.$$

One can estimate the probability that a random number near p divides L using Dickman's function. The reciprocal of this probability is the approximate average number of elliptic curves needed to factor n.

Example of factoring by ECM

Let $n = 2773$ and $P = (1, 3)$ be on the curve

$$y^2 \equiv x^3 + 4x + b \pmod{2773}.$$

Then b must be $3^2 - 1^3 - 4(1) = 4$.

Compute $2P = P + P$. The slope of the tangent line is

$$s = \frac{3 \cdot 1^2 + 4}{2 \cdot 3} = \frac{7}{6}.$$

The Euclidean algorithm finds $\gcd(6, 2773) = 1$ and gives $2311 \cdot 6 \equiv 1 \pmod{2773}$, so the slope is $s = 7 \cdot 2311 \equiv 2312 \pmod{2773}$. Then

$$x_3 \equiv 2312^2 - 1 - 1 \equiv 1771 \pmod{2773},$$

$$y_3 \equiv 2312(1 - 1771) - 3 \equiv 705 \pmod{2773}.$$

Thus $2P = P + P = (1771, 705)$.

Example of factoring by ECM, continued

Now let us compute $3P = 2P + P = (1771, 705) + (1, 3)$. The slope is

$$\frac{705 - 3}{1771 - 1} = \frac{702}{1770}.$$

We try to invert 1770 modulo 2773, but the Euclidean algorithm gives $\gcd(1770, 2773) = 59$. Therefore, we can't invert 1770 modulo 2773. But we don't care since we have factored $2773 = 59 \cdot 47$, which was our goal.

Here is what happened: Think of the curve modulo 2773 as a curve modulo 59 and a curve modulo 47. We have $3P = \infty \pmod{59}$ but $3P \neq \infty \pmod{47}$. (In fact, $4P = \infty \pmod{47}$, so if we had tried to compute $4P = 2(2P)$, we would have found the factor 47.)

Finding multiples of P on this elliptic curve makes the two factors of 2773 behave differently, and lets us separate them.

Elliptic curve prime proving

It is a variation of the Pocklington-Lehmer theorem, which says:

Pocklington-Lehmer Theorem: Let $n > 1$ be an integer. Let $n - 1 = rs$, with $r \geq \sqrt{n}$. Suppose that for each prime q dividing r, there is an integer a_q with $a_q^{n-1} \equiv 1 \pmod{n}$ and

$$\gcd(a_q^{(n-1)/q} - 1, n) = 1.$$

Then n is prime.

Goldwasser-Kilian Theorem: Let $n > 1$ be an integer. Let E be an elliptic curve modulo n. Suppose there are distinct primes $q_1, \ldots, q_k$ and points $P_i \neq \infty$ on E such that $q_i P_i = \infty$ for $1 \leq i \leq k$ and

$$\prod_{i=1}^{k} q_i > (n^{1/4} + 1)^2.$$

Then n is prime.

Elliptic Curve Discrete Logarithms

The **discrete logarithm problem for elliptic curves** is to find an integer x for which $Q = xP$, where P and Q are two given points on an elliptic curve E modulo p.

Shanks' baby-step-giant-step algorithm can be easily modified to solve the discrete logarithm problem for elliptic curves. The index calculus algorithm for discrete logarithms works only in the group R_p integers of modulo p and is much faster than Shanks' method. Shanks' algorithm and some other algorithms due to Pollard are about the best one can do in elliptic curve groups.

Hence, the group R_p must be much larger than an elliptic curve group to achieve the same security. A rough rule of thumb is that R_p with a 1024-bit prime p is about as safe as an elliptic curve modulo a 128-bit prime.

Here is Shanks' baby-step-giant-step algorithm for solving the discrete logarithm problem $xP = Q$ in an elliptic curve E modulo a prime p in $O(\sqrt{p} \log p)$ time and $O(\sqrt{p})$ space.

Let $m = 1 + \lceil \sqrt{p} \rceil$. Compute and sort the m ordered pairs (j, mjP), for j from 0 to $m-1$, by the second coordinate (the point). Compute and sort the m ordered pairs $(i, Q - iP)$, for i from 0 to $m - 1$, by the second coordinate (the point). Find a pair (j, R) in the first list and a pair (i, R) in the second list. This search will succeed because every integer between 0 and $p + 1 + 2\sqrt{p}$ can be written as a two-digit number ji in base m. Finally, $x = mj + i$.

There is an elliptic curve variation of the Diffie-Hellman key exchange algorithm in which the group R_p is replaced by an elliptic curve. In it, Alice and Bob agree on an elliptic curve $E = E_{a,b}$ modulo a prime p and a point P of high order on E, perhaps a generator of the group. Let N be the order of the group. The group E and the point P need not be secret and Alice and Bob do not need to know N exactly. By Hasse's theorem, N is approximately p, and that approximation is good enough.

Alice secretly chooses a random x_A in $0 < x_A < N$ and computes $P_A = x_A P$ on E. Bob secretly chooses a random x_B in $0 < x_B < N$ and computes $P_B = x_B P$ on E.

Alice sends P_A to Bob. Bob sends P_B to Alice. An eavesdropper, knowing E and P, and seeing P_A and P_B, cannot compute x_A or x_B from this data unless he can solve the discrete logarithm problem for elliptic curves quickly.

Alice computes $K_A = x_A P_B$ on E. Bob computes $K_B = x_B P_A$ on E. Then

$$K_A = (x_A \cdot x_B)P = K_B.$$

Alice and Bob choose certain agreed-upon bits from K_A to use as their key for a private key cipher like DES or AES.

Since the discrete logarithm problem is harder to solve for an elliptic curve than for the multiplicative group of integers modulo p, the modulus of the elliptic curve may be chosen smaller than the prime p for R_p.

The elliptic curve Pohlig-Hellman cipher works just like the Pohlig-Hellman cipher except that the multiplicative group R_p of integers modulo p is replaced by an elliptic curve.

Let p be a large prime and let E be an elliptic curve modulo p that has order N, that is, E has N points including the identity ∞.

We will explain shortly how a plaintext block M might be embedded into the x-coordinate of a point P on E. Assume this has been done.

A point P on E is enciphered by adding it to itself e times, using fast multiplication; the ciphertext point is $Q = eP$. The latter is deciphered by multiplying by d: $P = dQ$.

In order for the deciphering to return to P, the multipliers e and d must satisfy $ed \equiv 1 \pmod{N}$, because $NP = \infty$, and so $cP = P$ for any $c \equiv 1 \pmod{N}$. This implies that e (and d) must be chosen relatively prime to N.

By Hasse's theorem, N is approximately p. But this approximation is not good enough. We must know N exactly in order to choose e and d. In typical cryptographic applications, N and p must be large enough so that only Schoof's algorithm is fast enough to compute N. If you have a program for Schoof's algorithm, then you are free to choose any elliptic curve E for the analogue of the Pohlig-Hellman cipher. Otherwise, you must choose an elliptic curve whose order has been published.

In a known-plaintext attack on this system, one is given E, p, N (which are public anyway), P and Q, and one must find e with $Q = eP$ or, equivalently, d with $P = dQ$. Either problem is the discrete logarithm problem on the elliptic curve E, whose solution was just discussed.

We now explain how to embed plaintext into points. There are two methods in common use. Both embed a plaintext M in $0 < M < p$ into the x-coordinate of a point $P = (x, y)$ on a given elliptic curve E.

The first method is probabilistic and may fail to embed M with a small positive probability. The overall encryption function must handle this failure gracefully. It may

1. skip M,

2. change M in some way, or

3. ask for human assistance in changing M.

In any case, it is easy to make the probability of failure tiny.

Reserve k bits of the x-coordinate for a small integer. Then the blocks M must be k bits shorter, that is, $0 < M < p/2^k$ rather than $0 < M < p$. The probability of failure will be only 1 chance in 2^{2^k}. This is less than one chance in a billion if $k = 5$. The x-coordinate will be $x = 2^k M + i$, where i is a k-bit integer $0 \le i < 2^k$. When P is recovered during deciphering, M is extracted from x by $M = \lfloor x/2^k \rfloor$, which may be done quickly with a right shift of x by k bits. Let the elliptic curve have equation $y^2 \equiv x^3 + ax + b \pmod{p}$. Choose i by this algorithm:

```
for (i=0 to 2^k-1) {
    x=2^kM+i
    if (((x^3+ax+b)/p)=+1) { return i }
    }
return "Failure: could not choose i"
```

The algorithm returns the first $i < 2^k$, if any, for which the Legendre symbol $((x^3 + ax + b)/p) = +1$. Since the Legendre symbol (r/p) is $+1$ for $(p-1)/2$ values of r modulo p, and since, for each i, the value $x^3 + ax + b$ is more or less random modulo p, the probability that all 2^k choices for i yield $((x^3+ax+b)/p) \neq +1$ is about 2^{-2^k}, as claimed.

Once we have i with $((x^3 + ax + b)/p) = +1$, where $x = 2^k M + i$, we find a square root y of x modulo p by the methods of Chapter 7 and let $P = (x, y)$. Then P lies on E.

The second method of embedding plaintext into points is deterministic but only works for special primes p and elliptic curves E. Plenty of primes and elliptic curves satisfy the requirements.

Assume that $p \equiv 3 \pmod 4$. For such primes p, -1 is a quadratic nonresidue, so $(-1/p) = -1$. Let $b = 0$ in the congruence defining E, so that E is $y^2 \equiv x^3 + ax \pmod p$.

Plaintext M is restricted to $0 < M < p/2$. One bit of possible plaintext storage space is lost. Given M, form $t = M^3 + aM \bmod p$. Since $(-1/p) = -1$, exactly one of t and $-t$ is a quadratic residue modulo p. If $(t/p) = +1$, let $x = M$. If $(t/p) = -1$, let $x = p - M$. Then $((x^3 + ax)/p) = +1$ and we can find y with $y^2 \equiv x^3 + ax \pmod p$. Let $P = (x, y)$. When P is recovered during deciphering, look at x. If $x < p/2$, then $M = x$. If $x > p/2$, then $M = p - x$.

Elliptic Curve Massey-Omura cipher

Consider an elliptic curve Pohlig-Hellman cipher with elliptic curve E having N points modulo a prime p. Suppose users A and B have encryption functions E_A and E_B and decryption functions D_A and D_B. (So $E_A(P) = e_A P$ on E. $D_A(Q) = d_A Q$ on E, where $e_A d_A \equiv 1 \pmod{N}$, etc.) Since the encryption and decryption functions are all multiplication of integers times points on E, they may be done in any order and give the same result. For example, $E_A(D_B(P)) = D_B(E_A(P))$ for every P because both are just $e_A d_B P = d_B e_A P$.

How do A and B use this property as a public-key cipher? The "public key" consists of E, N, and p. The private keys are *all* of the multipliers e_A, d_A, etc. If Alice wants to send a message $0 < M < p$ to Bob, she first embeds it in a point P of E, as explained above. She sends $E_A(P)$ to Bob. Bob replies by sending $E_B(E_A(P))$ to Alice. Then Alice sends $D_A(E_B(E_A(P))) = E_B(D_A(E_A(P))) = E_B(P)$ to Bob. Bob deciphers the message by applying D_B to $E_B(P)$.

An eavesdropper would see the messages

$$R = E_A(P),$$

$$S = E_B(E_A(P))$$

$$T = D_A(E_B(E_A(P))) = E_B(P)$$

pass between Alice and Bob. If the eavesdropper could solve the discrete logarithm problem for points of E, then he could read P in either of two ways. First, $S = e_B R$. He knows S, R, E, N, and p. If he can solve for e_B, then he can compute d_B by the extended Euclidean algorithm. Then he can compute $P = d_B T$. The other way to read P is to use the equation $S = e_A T$ to find e_A, by solving a different discrete logarithm problem on E. Then compute d_A from e_A by the extended Euclidean algorithm and find $P = d_A R$. It is likely that the two discrete logarithm problems are equally hard.

Recall the ElGamal cipher

Say B wants to send a secret message M to A. The organizer of the system chooses a large prime p and an integer a_0 modulo p. These two numbers are public.

To use the system, A chooses a secret integer a_A modulo p and computes the integer $B_A = a_0^{a_A} \bmod p$. A makes public B_A and keeps a_A secret.

To sent M to A, B chooses a secret integer k and computes y_1 and y_2 by

$$y_1 \equiv a_0^k \qquad y_2 \equiv M B_A^k \pmod{p}.$$

B sends (y_1, y_2) to A.

A deciphers (y_1, y_2) by computing

$$M = y_2 y_1^{-a_A} \pmod{p}.$$

Elliptic Curve ElGamal cipher

There is an elliptic curve analogue to the ElGamal public key cryptosystem defined as follows:

Fix an elliptic curve E modulo p and a point P_0 of large order on E. All of this data is public.

Each user A who wishes to participate in this public-key cryptosystem chooses a secret a_A in $0 < a_A < p$ and publishes $P_A = a_A P_0$ on E.

When a user B wants to send a secret message M to A, she first embeds M into a point P of E. B chooses a random k in $0 < k < p$ and sends to A the pair $C = (kP_0, kP_A + P)$.

A deciphers this pair by

$$P = (kP_A + P) - a_A(kP_0).$$

The plaintext P is enciphered by adding the point kP_A in the second component of C. Note that $kP_A = k(a_A P_0) = (ka_A)P_0$. The first component of C provides a hint for deciphering P from the second component of C, but one which is useful only to A. Only A knows the secret key a_A, so only A can compute $a_A(kP_0) = (ka_A)P_0$. If this point is subtracted from the second component, one recovers P: $kP_A + P - (ka_A)P_0 = (ka_A)P_0 + P - (ka_A)P_0 = P$.

An eavesdropper who could solve the discrete logarithm problem on E could compute P from C and public data without knowing a_A as follows. The first component of C is $P_1 = kP_0$. This and $T = kP_A + P$ are observed by the eavesdropper. The eavesdropper knows p, E and P_0 because this information is public. He can also obtain A's public key P_A from A's directory, just as B did. He would solve the elliptic curve discrete logarithm problem $kP_0 = P_1$ for k and then compute $T - kP_A = (kP_A + P) - kP_A = P$.

The Weil Pairing

It is very complicated. We give a simplified version and two applications of it to cryptography.

Let $p = 6q - 1$ be a prime for which q is also prime. (For example, $q = 5$, $p = 29$.)

Let E be the elliptic curve

$$y^2 \equiv x^3 + 1 \pmod{p}.$$

One can show that in this situation every integer modulo p has exactly one cube root modulo p and that E has exactly $p+1 = 6q$ points. (Such a curve is called supersingular.)

One can show that there is a point $P_0 \neq \infty$ on E with $qP_0 = \infty$. If P is on E, then $6P$ is almost certainly a multiple of P_0 and $\neq \infty$.

The Weil Pairing continued

One can show that there is a function $\tilde{e}$ that maps pairs of points (aP_0, bP_0) to q-th roots of unity (in the complex numbers).

It is bilinear, that is,

$$\tilde{e}(P, Q + R) = \tilde{e}(P, Q)\tilde{e}(P, R)$$

$$\tilde{e}(P + Q, R) = \tilde{e}(P, R)\tilde{e}(Q, R)$$

$$\tilde{e}(aP_0, bP_0) = \tilde{e}(P_0, P_0)^{ab}.$$

Given two points P, Q that are multiples of P_0, one can compute $\tilde{e}(P, Q)$ quickly from the coordinates of P and Q (without knowing a and b).

Finally, $\tilde{e}(P_0, P_0) \neq 1$.

$\tilde{e}$ a (modified) Weil pairing.

The Weil Pairing continued

Identity-based encryption uses a users' login id as public key.

It uses two public hash functions.

H_1 maps arbitrary bit strings to multiples of P_0, shown as their coordinates (x, y), not as kP_0. Given a bit string b, one should not be able to find k with $H_1(b) = kP_0$.

H_2 maps q-th roots of unity to bit strings of length $n =$ the length of messages to be sent. (So $p > 2^n$.) (Make $n > 256$, the size of an AES key.)

The Weil Pairing continued

Trent, the Trusted Authority, sets up the system by choosing a random secret integer s as key. He never reveals s.

Trent computes and publishes $P_1 = sP_0$.

Bob (`bob@purdue.edu`) learns a secret from Trent to help decipher messages sent to him. The secret is $D_{Bob} = sH_1(\texttt{bob@purdue.edu})$. Trent does not save this point of E. Bob saves it.

If Bob loses D_{Bob}, he could ask Trent for it again. Trent won't give D_{Bob} to anyone other than Bob.

The Weil Pairing continued

Say Alice wants to send the message M of length n bits to Bob.

She knows Bob's email `bob@purdue.edu` and computes

$$g = \tilde{e}(H_1(\texttt{bob@purdue.edu}), P_1).$$

g is a q-th root of unity.

Alice chooses a random integer $r \not\equiv 0 \pmod{q}$ and computes $t = M \oplus H_2(g^r)$, a bit string of length n.

Alice sends Bob the ciphertext $C = (rP_0, t)$.

The Weil Pairing continued

Bob receives ciphertext $C = (U, v)$, where U is a point on E and v is a bit string of length n.

He computes $h = \tilde{e}(D_{Bob}, U)$, a q-th root of unity. He finds the plaintext by $M = v \oplus H_2(h)$.

In case $U = rP_0$ and $v = m \oplus H_2(g^r)$, Bob computes

$$h = \tilde{e}(sH_1(\texttt{bob@purdue.edu}), rP_0) =$$

$$= \tilde{e}(H_1(\texttt{bob@purdue.edu}), sP_0)^r = g^r,$$

so

$$t \oplus H_2(h) = t \oplus H_2(g^r) = M \oplus H_2(g^r) \oplus H_2(g^r) = M.$$

The Weil Pairing continued

The tripartite Diffie-Hellman key exchange takes only one round for Alice, Bob and Chuck to agree on a common AES key. (The normal Diffie-Hellman key exchange would take two rounds to do this.)

Same setup as before.

Alice, Bob and Chuck choose secret integers a, b, c, respectively.

Alice broadcasts (sends to both Bob and Chuck) aP_0. Bob broadcasts bP_0. Chuck broadcasts cP_0.

Alice computes $\tilde{e}(bP_0, cP_0)^a$. Bob computes $\tilde{e}(aP_0, cP_0)^b$. Chuck computes $\tilde{e}(aP_0, bP_0)^c$.

All three have computed $\tilde{e}(P_0, P_0)^{abc}$. They choose the low-order 128 bits as their common key.