CS 302 Computer Security and Privacy

Debayan Gupta

Lecture 7
February 19, 2019

Elliptic Curves Basics

Elliptic Curve Cryptography

Elliptic Curves Basics

Elliptic Curves

An *elliptic curve* E over a field K is a set of points (x, y) with $x, y \in K$, together with a special point $\mathcal{O}$ called the *point at infinity*. The (x, y) points are the roots of a Weierstrass equation of the form:

$$E: y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6,$$

where the polynomial on the right hand side has no double roots.

For particular fields K, the Weierstrass equation takes a simpler form, as we shall see.

EC over Real Numbers

Definition

Let $a, b \in \mathbb{R}$ be constants such that $4a^3 + 27b^2 \neq 0$.
A *non-singular* elliptic curve is the set E of solutions $(x, y) \in \mathbb{R} \times \mathbb{R}$ to the equation

$$y^2 = x^3 + ax + b$$

together with a special point $\mathcal{O}$ called the *point at infinity*.

The point at infinity is sometimes denoted by ∞.

EC over Real Numbers

The condition $4a^3 + 27b^2 \neq 0$ is necessary and sufficient to ensure that the elliptic curve equation does not have repeated roots.

If $4a^3 + 27b^2 = 0$, then corresponding elliptic curve is called a *singular* elliptic curve.

Singular elliptic curves are not safe for cryptographic uses.

Example

Each choice of the numbers yields a different elliptic curve.

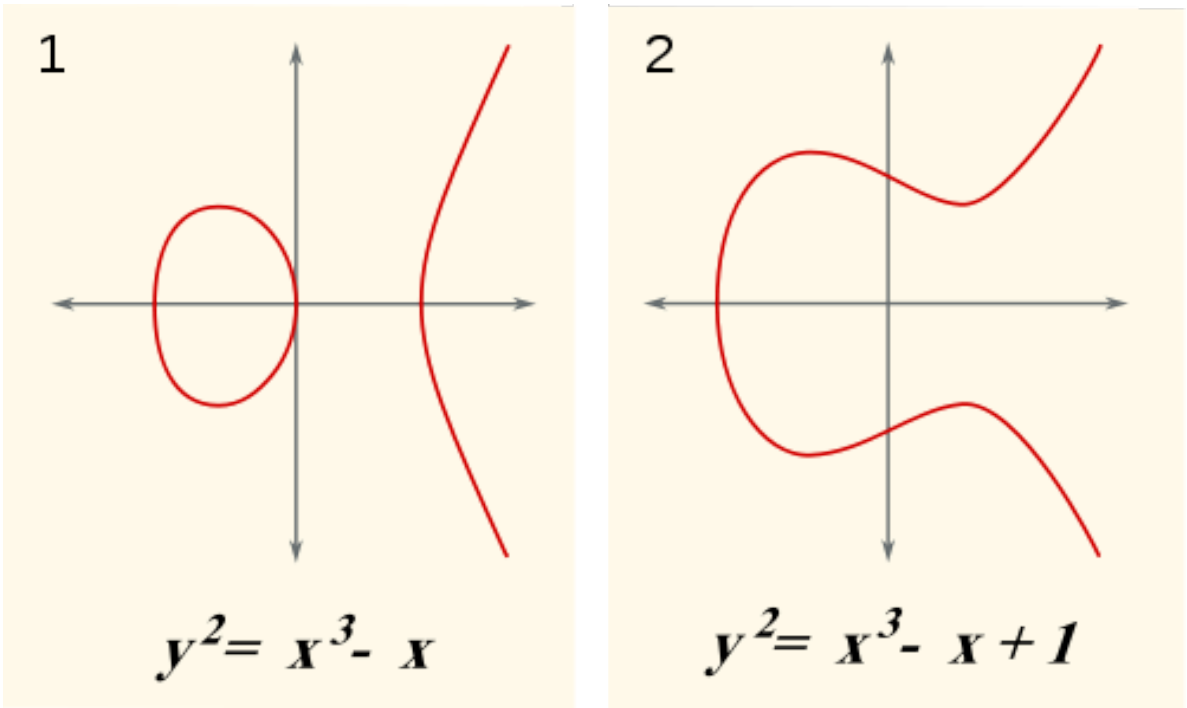


Image retrieved from http://en.wikipedia.org/wiki/Elliptic_curve

EC Operations

Exactly one of these conditions holds for any pair of points on an elliptic curve.

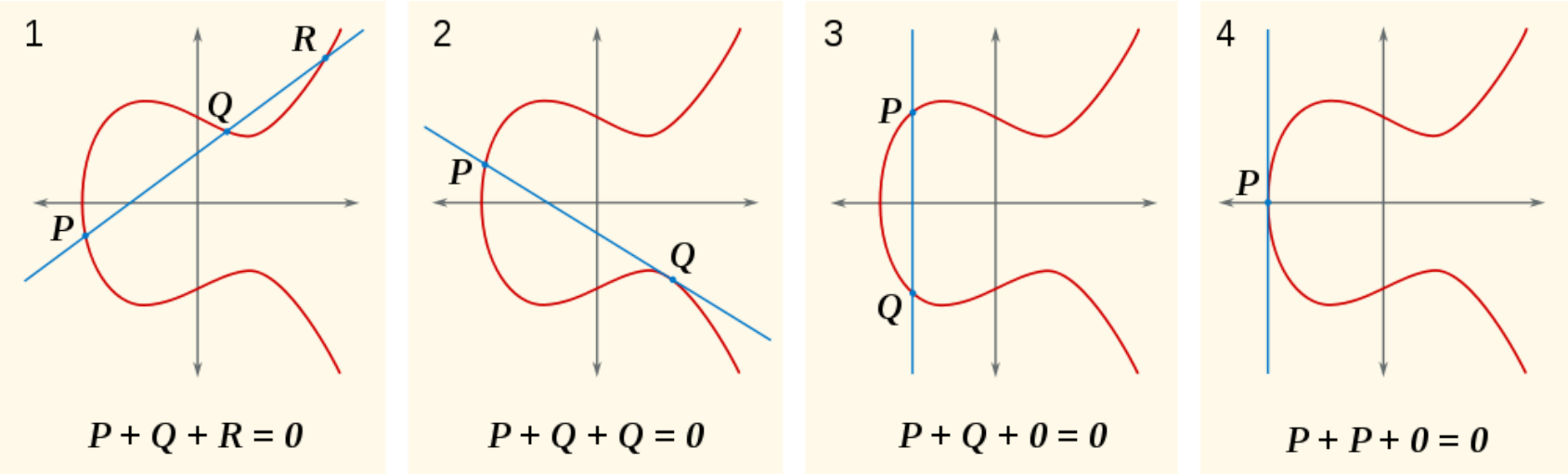


Image retrieved from http://en.wikipedia.org/wiki/Elliptic_curve

EC Operations

"Addition of Points on an Elliptic Curve over the Reals" from the Wolfram Demonstrations Project.

Contributed by Eric Errthum

EC Modulo a Prime

For prime p, the integers in $\mathbf{Z}_p$ form a field $\mathbb{F}_p$, so every non-zero element has a multiplicative inverse modulo p.

Elliptic curves over $\mathbb{F}_p$ are defined exactly as they are over real numbers but with all arithmetic being performed over $\mathbb{F}_p$.

Definition

Let $p > 3$ be a prime. The elliptic curve $y^2 = x^3 + ax + b$ over $\mathbb{F}_p$ is the set of solutions $(x, y) \in \mathbb{F}_p \times \mathbb{F}_p$ to the congruence

$$y^2 \equiv x^3 + ax + b \pmod{p},$$

together with a special point $\mathcal{O}$ called the *point at infinity*.
$a, b \in \mathbb{F}_p$ are constants such that $4a^3 + 27b^2 \not\equiv 0 \pmod{p}$.

Example

$E : y^2 = x^3 - x$ over a finite field $\mathbb{F}_{61}$

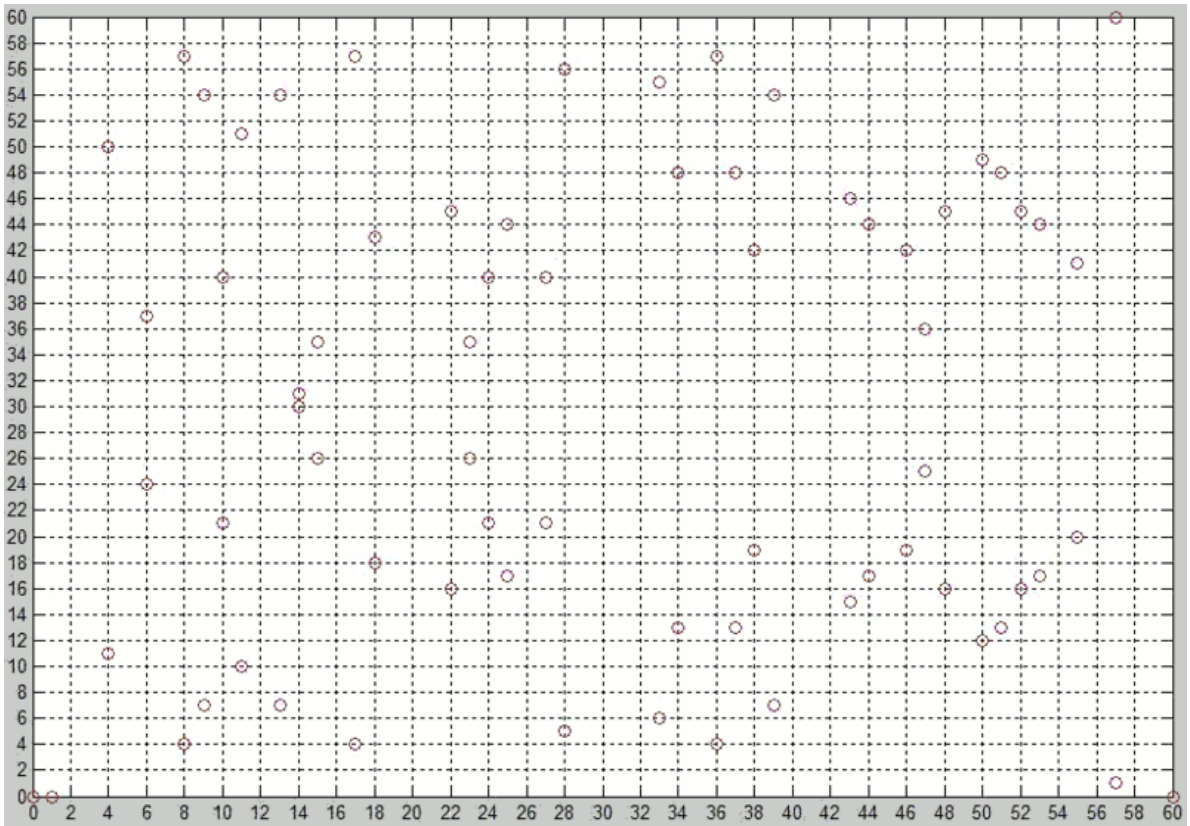


Image retrieved from http://en.wikipedia.org/wiki/Elliptic_curve

Example

A 3D graph of an elliptic curve $E : y^2 = x^3 + 673x$ over $\mathbb{F}_{677}$.

URL: `http://www.youtube.com/watch?v=QFLQWhvdIYU`

Example

Elliptic curves mod p are finite sets of points. These are the elliptic curves we are interested in.

- How can we find those points?
- How many points are on an elliptic curve?

Example

$E : y^2 \equiv x^3 + 4x + 4 \pmod 5$ creates the following group:

$$(0,2), (0,3), (1,2), (1,3), (2,0), (4,2), (4,3), \mathcal{O}$$

Finding points:

- Substitute each possible value of $x = \{0, 1, 2, 3, 4\}$ into the equation and find the values of y that solve the equation.
- For example, $x \equiv 0 \Rightarrow y^2 \equiv 4 \Rightarrow y \equiv 2, 3 \pmod 5$, which gives us two points $(0, 2)$ and $(0, 3)$.

y	0	1	2	3	4
$y^2 \bmod 5$	0	1	4	4	1

Number of Points on a Curve

Theorem (Hasse's Theorem)

Suppose E mod p *has N points. Then*

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

Hasse's theorem bounds the number of points on an elliptic curve over a finite field.

$\#E(\mathbb{F}_p)$ lies in the interval $[p+1-2\sqrt{p}, p+1+2\sqrt{p}]$.

Adding Points

All arithmetic operations are performed in $\mathbb{F}_p$.

Unfortunately, the addition of points on an elliptic curve over $\mathbb{F}_p$ does not have the nice geometric interpretation that it does on an elliptic curve over $\mathbb{R}$.

Adding Points

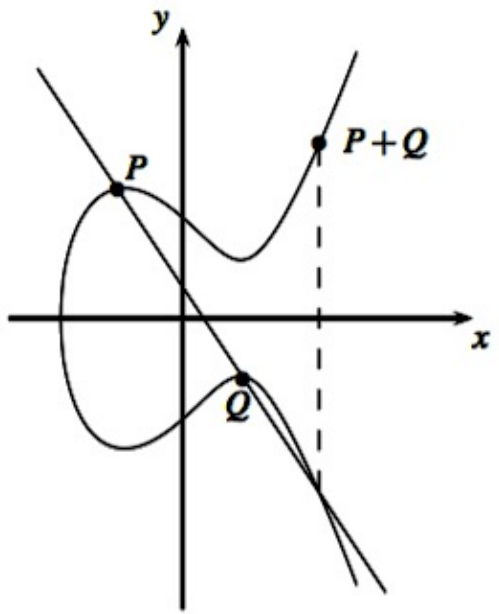
Let $P = (x_P, y_P)$, $Q = (x_Q, y_Q)$ and $R = (x_R, y_R)$ be points on E.

1. Add the point at infinity to itself.
   $\mathcal{O} + \mathcal{O} = \mathcal{O}$
2. Add the point at infinity to any other point.
   $P + \mathcal{O} = \mathcal{O} + P = P$
3. Add two points with the same x-coordinates and different (or equal to 0) y-coordinates: $x_Q = x_P$ and $y_Q = -y_P$.
   $P + Q = \mathcal{O}$

Adding Points

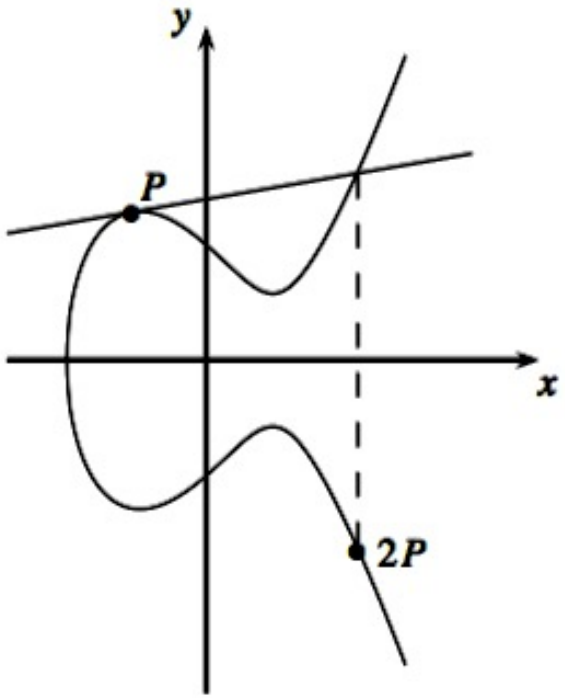
4. Add two points with different x-coordinates.

$$P + Q = R$$
$$x_R = \lambda^2 - x_P - x_Q$$
$$y_R = \lambda(x_P - x_R) - y_P$$
$$\lambda = (y_Q - y_P)(x_Q - x_P)^{-1}$$



Adding Points

5. Add a point to itself (*point doubling*).
   $P + P = R$
   $x_R = \lambda^2 - 2x_P$, $y_R = \lambda(x_P - x_R) - y_P$
   $\lambda = (3x_P^2 + a)(2y_P)^{-1}$



Why adding points works?

The algebraic formula follows the geometric addition. Here is how it works.

To add two points, P and Q, we define L to be the line through P and Q. The line L will intersect E in one further point R'. If we reflect R' in the x-axis, then we get a point which we name R.

The equation of L is $y = \lambda x + \nu$, where the slope of L is

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

and

$$\nu = y_P - \lambda x_P = y_Q - \lambda x_Q$$

Why adding points works?

In order to find the points $E \cap L$, we substitute $y = \lambda x + \nu$ into the equation E, obtaining the following:

$$(\lambda x + \nu)^2 = x^3 + ax + b$$

which is the same as

$$x^3 - \lambda^2 x^2 + (a - 2\lambda\nu)x + b - \nu^2 = 0$$

$E \cap L$ consists of three points, two of which we already know: P and Q. The roots of the above equation are the x-coordinates of the points in $E \cap L$, hence, x_P and x_Q are the two roots.

Why adding points works?

Since the equation is cubic, there are three roots. The sum of three roots must be the negative of the coefficient of the quadratic term, or λ^2. Therefore:

$$x_{R'} = \lambda^2 - x_P - x_R$$

where $x_{R'}$ is the x-coordinate of the point R'. We will denote the y-coordinate of R' by $-y_R$, so the y-coordinate of R will be y_R.

An easy way to compute y_R is to use the fact that the slope of L, namely λ is determined by any two points on L.

Why adding points works?

If we use the points (x_P, y_P) and $(x_R, -y_R)$ to compute this slope, we get:

$$\lambda = \frac{-y_R - y_P}{x_R - x_P}$$

or

$$y_R = \lambda(x_P - x_R) - y_P$$

Note, that $x_{R'} = x_R$. Therefore, we derived a formula for $P + Q = R$ if $P \neq Q$. A formula for $P + Q = R$ if $Q = P$ can be derived in a similar fashion.

EC Groups

As noted before, elliptic curves mod p are finite sets of points.

The set of points on E forms a group given the $+$ operator. The group operator is defined using the addition law.

The group is abelian since $P + Q = Q + P$.

Notation:
$E(\mathbb{F}_p)$ denotes an elliptic curve group over $\mathbb{F}_p$.
$\#E(\mathbb{F}_p)$ denotes the order (cardinality) of $E(\mathbb{F}_p)$.

Why points over an EC form a group?

Definition

A group $(G, \circ)$ is a set G with a binary operation $\circ : G \times G \to G$ such that the following three axioms are satisfied:

Associativity: For all $a, b, c \in G$ the equation $(a \circ b) \circ c = a \circ (b \circ c)$ holds.

Identity element: There is an element $e \in G$ s.t. for all $a \in G$ the equation $e \circ a = a \circ e = a$ holds.

Inverse element: For each $a \in G$ there exists an element $b \in G$ s.t. $a \circ b = b \circ a = e$.

Why points over an EC form a group?

Q: Does it really work?

Associativity: $(P + Q) + Z \stackrel{?}{=} P + (Q + Z)$

Identity element: What is it?

Inverse element: What is it?

Why points over an EC form a group?

Associativity: Points can be added in any order.

Identity element: $\mathcal{O}$ is an identity with respect to addition.

Inverse element: Every point on E has an inverse with respect to addition: $P + (-P) = \mathcal{O}$ where $P = (x_P, y_p)$ and $-P = (x_P, -y_P)$.

Therefore, $(E, +)$ is a group.

Additionally, the group operator $+$ is commutative since $P + Q = Q + P$. Hence, $(E, +)$ in an abelian group.

Other Operations

For many of the crypto schemes we need to perform multiplication. In our case we have the $+$ operator to work with.

Let k be an integer and P a point on E. $k \times P$ (or kP)[2] is defined as adding P to itself k times.

Once we calculate $k \times P$, it is extremely difficult to recover k from $k \times P$. The only way to recover k from $k \times P$ is to try every possible repeated addition of P.

Q: Does it sound familiar?

[2] Note that we do not define a multiplication operator over E.

Elliptic Curve Discrete Logarithm Problem

Let P be a point on E. Compute $Q = k \times P$. Then, ECDLP: given P and Q compute k.

This allows us to translate crypto schemes based on DLP to EC-based schemes.

Elliptic Curve Cryptography

Elliptic Curve Cryptography

Originally independently proposed by Neal Koblitz (University of Washington) and Victor Miller (IBM) in 1985.

ECC was proposed as an alternative to other public key encryption algorithms, for example RSA.

All ECC schemes are public key and are based on ECDLP.

EC Cryptosystems

There are many EC cryptosystems used in practice. We will have a look at three elliptic curve versions of classical cryptosystems:

1. Diffie-Hellman Key Exchange
2. ElGamal
3. DSA

Why ECC?

The computational overhead of RSA increases with the key length. Faster computers and better factorization algorithms force us to use longer keys.

In case of EC, we are able to use smaller primes, or smaller finite fields, and achieve a level of security comparable to that for much larger integers mod p.

This allows for much more efficient cryptosystems!

Comparison of Key Lengths

Symmetric Key Size (bits)	RSA and Diffie-Hellman Key Size (bits)	Elliptic Curve Key Size (bits)
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

Image retrieved from http://www.nsa.gov/business/programs/elliptic_curve.shtml

Note: The above URL no longer works, and I have been unable to find a replacement URL. See Elliptic curve cryptography: The serpentine course of a paradigm shift for a historical account for how elliptic curve cryptography gained acceptance over many years.

Where EC Cryptosystems are used?

EC Cryptosystems can be used wherever classic cryptosystems are used.

The main advantage of ECC are lower computational requirements. For this reason, ECC algorithms can be easily implemented on smart cards, pagers, or mobile devices. Some smart cards can only work with ECC.

ECC are also well suited for applications that need long term security requirements at a reasonable computational cost.

Changing a Classical Cryptosystem into EC System

There is a general procedure for changing a classical system based on discrete logarithms into one using elliptic curves:

1. Change modular multiplication to addition of points on an elliptic curve.
2. Change modular exponentiation to "multiplying" a point on an elliptic curve by an integer.

Representing Plaintext

In most cryptosystems, we need a way of mapping our message into a numerical value upon which we can perform mathematical operations.

To use EC cryptosystems, we need to map a message into a point on an elliptic curve.

Recall, that we can use a point on the curve and produce another point on the curve. EC cryptosystems use the plaintext point on E to yield a new point on E that will serve as a ciphertext.

Encoding Plaintext

The problem of encoding plaintext is quite difficult since there is no known polynomial time deterministic algorithm for writing down points on an arbitrary elliptic curve E mod p.

However, there are fast probabilistic methods for finding points and these can be used for encoding messages.

These methods have the property that with small probability they will fail to produce a point, however, by appropriately choosing parameters, this probability can be made arbitrarily small.

Example: Koblitz's Method (see 16.2.3 of Trappe & Washington)

Koblitz's Method

Main idea: embed a message m represented as a number into the x-coordinate of a point on E.

Because the probability that $m^3 + am + b$ is a square mod p is $\frac{1}{2}$, we add a few bits at the end of m and adjust them until we get a square.

The probability that we will fail to find a square (and hence fail to associate m with a point) is about $\frac{1}{2^k}$.

Koblitz's Method

Encoding

1. Choose an auxiliary base parameter k and verify that m satisfies $(m+1)k < p$.
2. The message m is represented by $x = mk + j$, where $0 \leq j \leq k$
3. For $j = 0, 1, 2, \ldots, k-1$, compute $x^3 + ax + b$ and solve for y.
4. If there is a square root y, then $P_m = (x, y)$, otherwise, increment j and try again.

Decoding

1. Compute $m' = \frac{x}{k}$ and set m to be the greatest integer $\leq m'$.

Koblitz's Method Example

Encode

1. Assume that the curve parameters are $p = 179$, $a = 2$, $b = 7$, $k = 10$.
2. The message to encode is $m = 5$.
3. First, check $x = mk + 0$. If you can't solve for y, check $x = mk + 1$, $x = mk + 2$, and so on.
   $x = 5 * 10 = 50$, no y exists
   $x = 5 * 10 + 1 = 51$,
   $y = 51^3 + 2 * 51 + 7 = 121 = 11 \bmod 179$.
4. Create $P_m = (51, 11)$.

Decode

1. Compute $\frac{x}{k} = \frac{51}{10} = 5.1$.
2. Return 5 as the original plaintext.

EC Domain Parameters

EC Domain Parameters yield a set of information for communication parties to identify a certain elliptic curve group.

The domain parameters comprise:

- finite field $\mathbb{F}_p$
- coefficients a and b of the Weierstrass equation
- base point $G \in E(\mathbb{F}_p)$
- order of G
- cofactor $h = \frac{\#E(\mathbb{F}_p)}{n}$, where n is the order of G

Diffie-Hellman

Alice and Bob want to exchange a key. In order to do so, they agree on a prime p and a generator g.

1. Alice and Bob choose random integers k_A and k_B respectively.
2. Alice computes $A = g^{k_A}$ and sends to Bob.
3. Bob computes $B = g^{k_B}$ and sends to Alice.
4. Alice and Bob compute their key:
   - Alice: $B^{k_A} = g^{k_B k_A}$
   - Bob: $A^{k_B} = g^{k_A k_B}$

EC Diffie-Hellman

Q: What are the rules of changing a classical cryptosystem into EC System?

EC Diffie-Hellman

Alice and Bob want to exchange a key. In order to do so, they agree on an elliptic curve E and a public base point G on E.

1. Alice and Bob choose random integers k_A and k_B respectively.
2. Alice computes $A = k_A \times G$ and sends to Bob.
3. Bob computes $B = k_B \times G$ and sends to Alice.
4. Alice and Bob compute their key as $A \times B = k_A \times k_B \times G$
   - Alice: $k_A \times B = k_A \times (k_B \times G)$
   - Bob: $k_B \times A = k_B \times (k_A \times G)$

ElGamal

Recall non-EC version:

1. Alice wants to send a message m s.t. $0 \leq m < p$ to Bob.
2. Bob chooses a large prime p and a primitive root α. He also chooses a secret integer a and computes $\beta \equiv \alpha^a \pmod{p}$.
3. Bob makes (p, α, β) his public key and keeps a secret.
4. Alice chooses a random k and computes y_1 and y_2, where $y_1 \equiv \alpha^k$ and $y_2 \equiv \beta^k m \pmod{p}$.
5. She sends (y_1, y_2) to Bob, who then decrypts by calculating $m \equiv y_2 y_1^{-a} \pmod{p}$.

EC ElGamal

1. Alice wants to send a message m to Bob.
2. Bob chooses an elliptic curve E mod p. He chooses a point α on E and a secret integer a. He computes $\beta = a \times \alpha$.
3. The points α and β are made public, while a is kept secret.
4. Alice expresses her message as a point M on E. She chooses a random k, computes $Y_1 = k \times \alpha$ and $Y_2 = M + k \times \beta$, and sends the pair (Y_1, Y_2) to Bob.
5. Bob decrypts by calculating $M = Y_2 - a \times Y_1$.

Difficulties with EC ElGamal

There are some practical difficulties in implementing an EC ElGamal cryptosystem.

Message expansion: ElGamal has a message expansion factor of two. The EC version has a message expansion factor of about four because each ciphertext consists of four elements.

Message encoding: The plaintext space consists of points on the curve E and there is no convenient method to deterministically generate points on E. Koblitz's method is one approach, but it might fail on some message values.

A Better Elliptic Curve ElGamal Algorithm

Below is an improved algorithm based on the idea of "blinding" that allows any message in $\mathbf{Z}_p^*$ to be encrypted.

EC ElGamal (improved version)

1. Alice wants to send a message $m \in \mathbf{Z}_p^*$ to Bob.
2. Bob chooses an elliptic curve E mod p. He chooses a point α on E and a secret integer a. He computes $\beta = a \times \alpha$.
3. The points α and β are made public, while a is kept secret.
4. Alice chooses a random k and computes $\gamma = k \times \beta$. She then computes $Y_1 = k \times \alpha$ and $Y_2 = mx_0 \bmod p$, where x_0 is the x-coordinate of γ. She sends the pair (Y_1, Y_2) to Bob.
5. Bob decrypts by calculating $\gamma = a \times Y_1$, letting x_0 be the x-coordinate of γ, and then calculating $m = Y_2 x_0^{-1} \bmod p$.

The green formulas show where this algorithm differs from the one presented before.

Use of blinding

Both versions compute a point $\gamma = k \times \beta$ for a randomly chosen k. In both versions, γ is used as a blinding factor. The difference is how it is used.

In the first algorithm, the point M, chosen to represent the real message m, is blinded by adding the point γ to it, giving $Y_2 = M + \gamma = M + k \times \beta$.

In the improved algorithm, the message m is directly blinded with x_0, which is the x-coordinate of the point γ. This gives $Y_2 = mx_0 \bmod p$. Note that Y_2 is now a number in $\mathbf{Z}_p^*$, not a point on the elliptic curve.

EC DSA

Alice wants to sign a message m which satisfies $0 \leq m \leq n$. She needs to choose a prime p and an elliptic curve E.

Alice computes the number of points n on E and chooses a point A on E.

Alice chooses her secret integer a s.t. $1 < a \leq n - 1$ and computes $B = a \times A$.

The public information is (p, E, n, A, B).

EC DSA: Signing

Alice does the following to sign a message m:

1. Chooses a random integer k with $1 \leq k < n$ and computes $R = k \times A = (x, y)$.
2. Computes $s \equiv k^{-1}(m - ax) \pmod{n}$.
3. Sends the signed message (m, R, s) to Bob.

EC DSA: Verification

Bob verifies the signature as follows:

1. Computes $V_1 = x \times B + s \times R$ and $V_2 = m \times A$.
2. Declares the signature valid iff $V_1 = V_2$.

The verification works because

$$\begin{aligned} V_1 &= x \times B + s \times R \\ &= xa \times A + k^{-1}(m - ax)(k \times A) \\ &= xa \times A + (m - ax) \times A \\ &= m \times A \\ &= V_2 \end{aligned}$$

Choosing Elliptic Curves

A list of elliptic curves recommended by NIST for cryptographic uses is specified in FIPS PUB 186-4 (Appendix D).

http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

NIST Suggested Curve P-192

Prime modulus p

6277101735386680763835789423207666416083908700390324961279

Order n

6277101735386680763835789423176059013767194773182842284081

Coefficient a

64210519 e59c80e7 0fa7e9ab 72243049 feb8deec c146b9b1

Coefficient b

3099d2bb bfcb2538 542dcd5f b078b6ef 5f3d6fe2 c745de65

The base point x **coordinate** G_x

188da80e b03090f6 7cbf20eb 43a18800 f4ff0afd 82ff1012

The base point y **coordinate** G_y

07192b95 ffc8da78 631011ed 6b24cdd5 73f977a1 1e794811

Additional Resources

ECC Tutorial, Certicom.

https://www.certicom.com/content/certicom/en/ecc-tutorial.html

Douglas Stinson, "Cryptography: Theory and Practice", Second Edition, 2002.

Wade Trappe and Lawrence C. Washington, "Introduction to Cryptography with Coding Theory", Second Edition, 2006.