MH4311 Cryptography

Lecture 19
Elliptic Curve Public Key Cryptosystems

Wu Hongjun

Lecture Outline

- Classical ciphers
- Symmetric key encryption
- Hash function and Message Authentication Code
- Public key encryption
- Digital signature
- Key generation, establishment and management
- Elliptic curve public key cryptosystems
- Introduction to other cryptographic topics

Recommended Reading

- CTP Section: Section 6.5, 7.4.3
- Wikipedia

  https://en.wikipedia.org/wiki/Elliptic-curve_cryptography

  https://en.wikipedia.org/wiki/Elliptic_curve

Introduction

- An elliptic curve over a finite field can be used to construct a finite group
- The discrete logarithm problem of the elliptic curve group is not vulnerable to Index Calculus Algorithm
  - In the ElGamal public key cryptosystem and the Diffie-Hellman key exchange algorithm,
    if the elliptic curve group is used to replace the multiplicative group Z_p^*, we can get much smaller key size

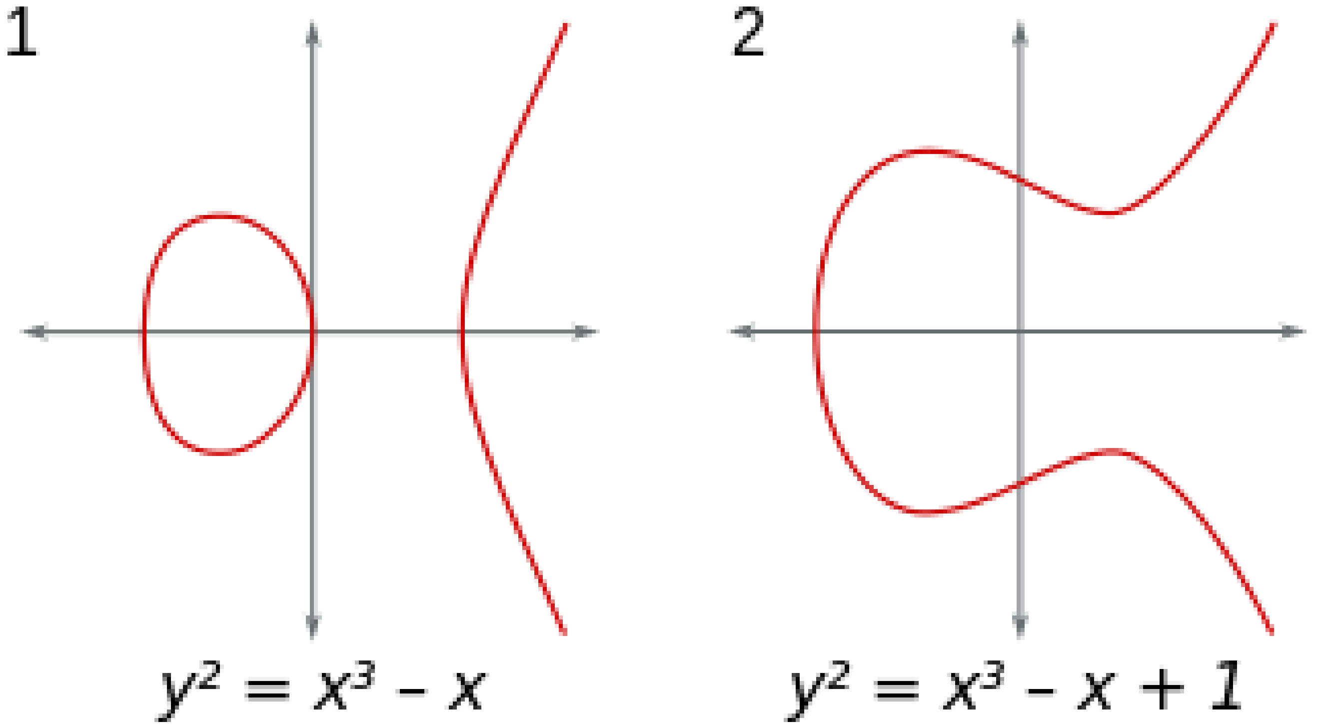
Elliptic Curve

Elliptic Curve over Real Number

- Elliptic curve over the real number is defined by an equation of the form
$$y^2 = x^3 + ax + b$$
where x, y, a and b are real numbers
(x and y are variables, a and b are constants)
  - For every solution (x, y) on the curve, (x, -y) is also a solution.

- Non-singular elliptic curve if $4a^3 + 27b^2 \neq 0$.
  - In this course, we consider only the non-singular elliptic curves

Elliptic Curve Examples



Source:
https://en.wikipedia.org/wiki/Elliptic_curve

Elliptic Curve over Real Number

- The points on the elliptic curve together with the "point at infinity" form a group E with the "addition" operation
  - the "point at infinity" is the point (∞, ∞), denoted as O
  - The point O is the identity element of the group

Elliptic Curve Addition

- The operation over the group E is addition "+" (different from the integer addition)
- Let P, Q be two elements in Group E.

$P = (x_1, y_1)$, $Q = (x_2, y_2)$

1) $P + O = O + P = P$

2) If $x_1 = x_2, y_1 = -y_2$, then $P + Q = O$, i.e., $P = -Q$.

P and Q are inverse of each other with respect to the elliptic curve addtition.

Elliptic Curve Addition

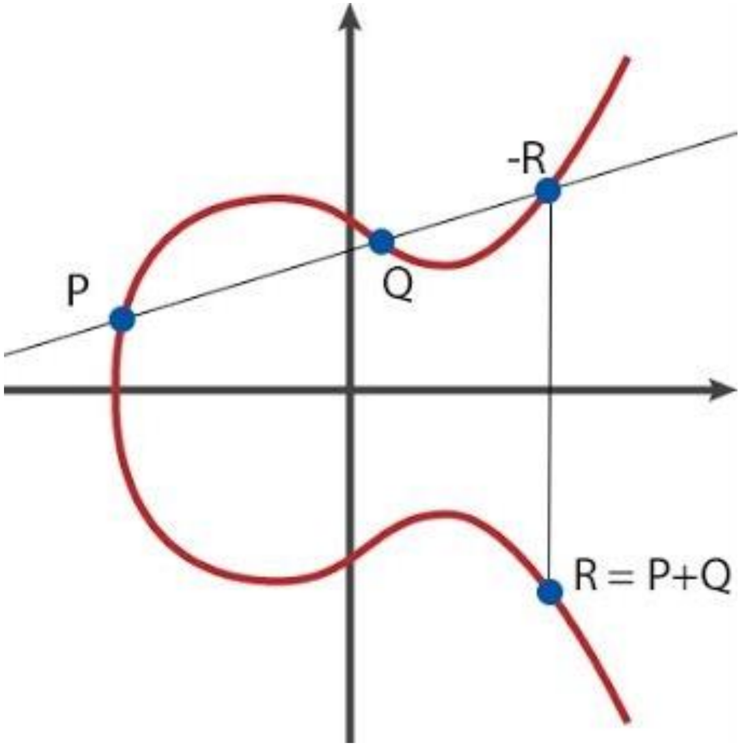
- Let P, Q, R be three elements in Group E.

  $P = (x_1, y_1)$, $Q = (x_2, y_2)$, $R = (x_3, y_3)$

  3) To compute $R = P + Q$, where P an Q are different points, and they are not on the same vertical line

  - Draw the line that intersects P and Q.
  - This line intersects the curve at a third point, $-R$.
  - We then take $P + Q$ to be the inverse of $-R$
  - The picture and formula are given on the next slide

Elliptic Curve Addition



3) $R = P + Q$. If $x_1 \neq x_2$, let $\lambda = \dfrac{y_2 - y_1}{x_2 - x_1}$,

then $x_3 = \lambda^2 - x_1 - x_2$,

$y_3 = \lambda(x_1 - x_3) - y_1$

Elliptic Curve Addition

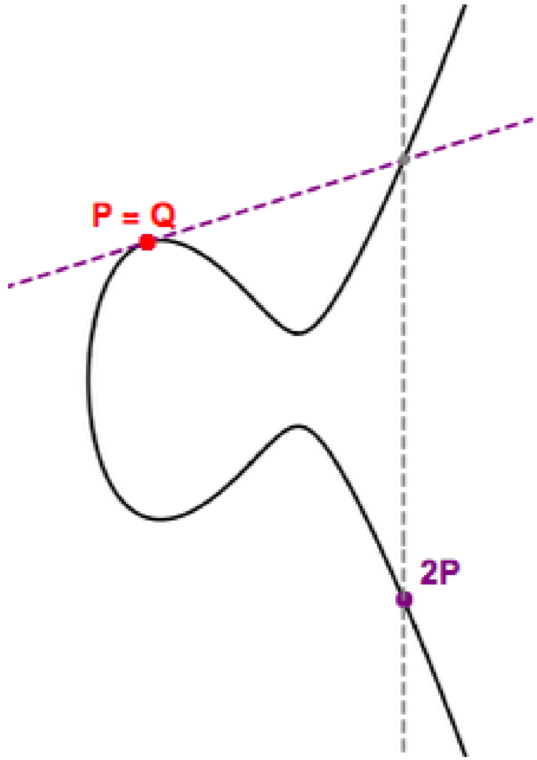
- Let P, Q, R be three elements in Group E.

  $P = (x_1, y_1)$, $Q = (x_2, y_2)$, $R = (x_3, y_3)$

  4) To compute $R = P + Q$, where P an Q are **the same point**

  - Draw the tangent line at point P
  - This line intersects the curve at a third point, $-R$.
  - We then take $P + P$ to be the inverse of $-R$
  - The picture and formula are given on the next slide

Elliptic Curve Addition



4) $R = P + Q$. If $x_1 = x_2, y_1 = y_2$, let $\lambda = \dfrac{3x_1^{\ 2} + a}{2y_1}$,

then $x_3 = \lambda^2 - x_1 - x_2$,

$y_3 = \lambda(x_1 - x_3) - y_1$

Elliptic Curve over Finite Field

Elliptic Curve over Finite Field

- Elliptic curve over a finite field is defined by an equation of the form

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

where *x*, *y*, *a* and *b* are elements of the finite field (*x* and *y* are variables, *a* and *b* are constants)

  - For every solution (*x*, *y*) on the curve, (*x*, -*y*) is also a solution.

- Non-singular elliptic curve if $4a^3 + 27b^2 \neq 0$ in the finite filed
  - We consider only the non-singular elliptic curves

Elliptic Curve over Finite Field

- Example:
  Elliptic curve $y^2 = x^3 + x + 1$ over GF(23)

We first check the value of $(4a^3 + 27b^2)$.

$$(4a^3 + 27b^2) \bmod 23 = 8 \neq 0$$

It means that the curve is non-singular.

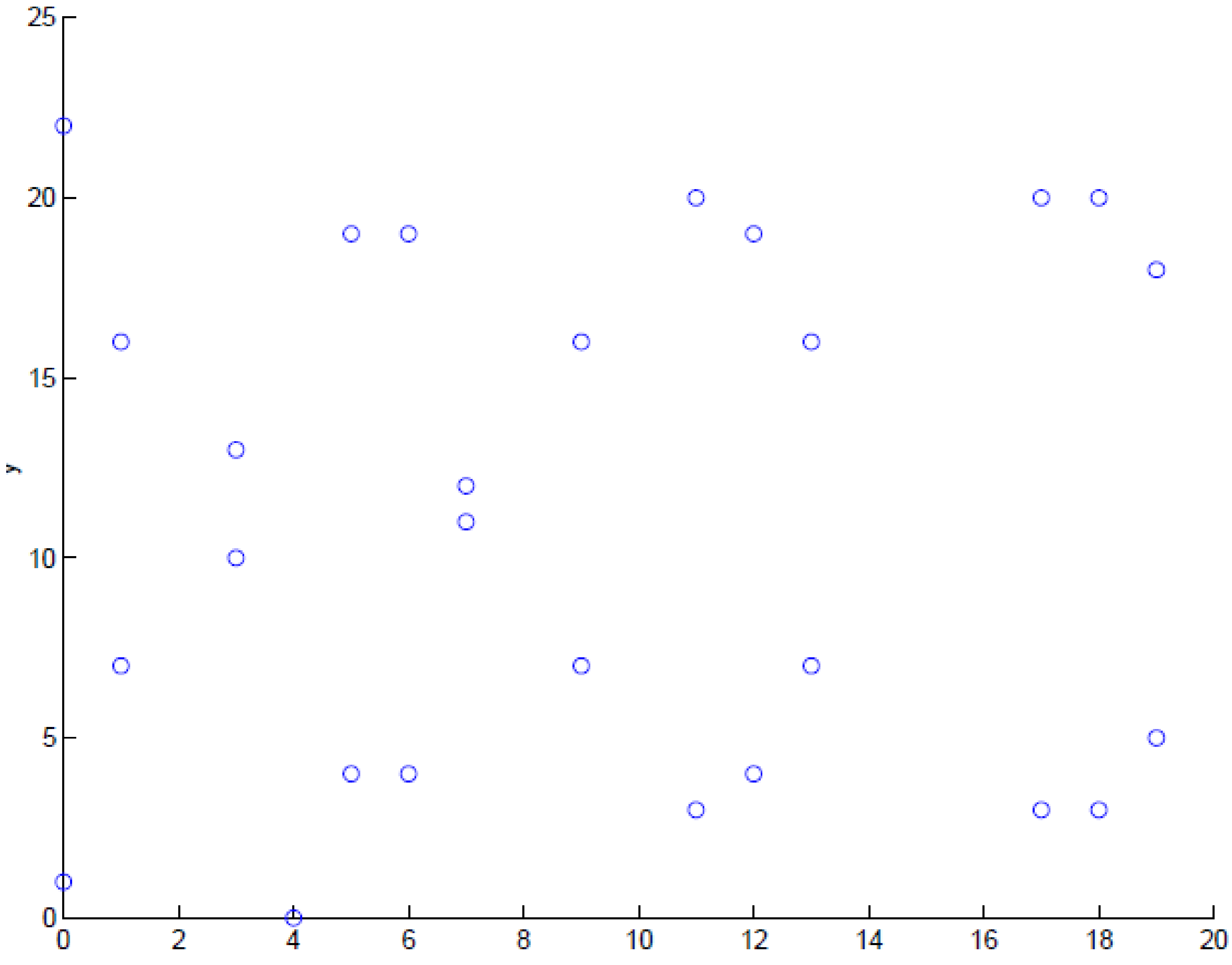
Elliptic Curve over Finite Field

- Example: (cont.)
  Elliptic curve $y^2 = x^3 + x + 1$ over GF(23)

Find all the points (x, y) on the curve:

$(0,1)$	$(0,22)$	$(1,7)$	$(1,16)$	$(3,10)$	$(3,13)$	$(4,0)$
$(5,4)$	$(5,19)$	$(6,4)$	$(6,19)$	$(7,11)$	$(7,12)$	$(9,7)$
$(9,16)$	$(11,3)$	$(11,20)$	$(12,4)$	$(12,19)$	$(13,7)$	$(13,16)$
$(17,3)$	$(17,20)$	$(18,3)$	$(18,20)$	$(19,5)$	$(19,18)$	

There is also "point at infinity" O
(identity element)



Elliptic curve $y^2 = x^3 + x + 1$ over GF(23)

Elliptic Curve over Finite Field

- The addition of the elliptic curve over a finite field uses the same formula as that of the elliptic curve over real number
- Example:
  Elliptic curve $y^2 = x^3 + x + 1$ over GF(23)
  - $P + O = P$ for all the points on the curve
  - Every point has an inverse
    The inverse of (x_1, y_1) is $(x_1, -y_1)$
    $(0, 1) + (0, 22) = O$
    $(18, 3) + (18, 20) = O$

Elliptic Curve over Finite Field

- Example: (cont.)
  Elliptic curve $y^2 = x^3 + x + 1$ over GF(23)

  $P = (3, 10)$, $Q = (18, 3)$. Compute $P + Q$.

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} = \frac{3-10}{18-3} \bmod 23$$

$$= 16 \times 15^{-1} \bmod 23 = 16 \times 20 \bmod 23 = 21$$

$$x_3 = \lambda^2 - x_1 - x_2 = (21^2 - 3 - 18) \bmod 23 = 6,$$

$$y_3 = \lambda(x_1 - x_3) - y_1 = (21 \times (3-6) - 10) \bmod 23 = 19$$

So $P + Q = (6,19)$

Elliptic Curve over Finite Field

- Example: (cont.)
  Elliptic curve $y^2 = x^3 + x + 1$ over GF(23)

  $P = (3, 10)$. Compute the value of $P + P = 2P$.

  $$\lambda = \frac{3x_1^2 + a}{2y_1} = \frac{3\times 3^2 + 1}{2\times 10} \bmod 23$$

  $$= 5\times 20^{-1} \bmod 23 = 5\times 15 \bmod 23 = 6$$

  $$x_3 = \lambda^2 - x_1 - x_2 = (6^2 - 3 - 3) \bmod 23 = 7,$$

  $$y_3 = \lambda(x_1 - x_3) - y_1 = (6\times(3-7) - 10) \bmod 23 = 12$$

  So $P + P = (7,12)$

Elliptic Curve over Finite Field

- The points on the elliptic curve over a finite field together with the "point at infinity" form a group E with the elliptic curve addition operation
  - In cryptography, the elliptic curve is over the finite field GF(p) or GF(2^m), where p is a prime at least 256-bit, m is an integer at least 256
  - The group of the elliptic curve over GF(p) is denoted as $\mathrm{E}_p(a, b)$, where a and b are the parameters of the curve
  - The group of the elliptic curve over GF(2^m) is denoted as $\mathrm{E}_{2^m}(a, b)$

Elliptic Curve Discrete Logarithm Problem

Elliptic Curve Discrete Logarithm Problem

- In the implementation of elliptic curve cryptosystem, it is important to find the order of the elliptic curve group E
  - The order of a group is the number of elements in the group
  - The order of an elliptic curve group is the number of points on the curve (together with the point at infinity)
  - The order of an elliptic curve group can be computed efficiently using Schoof's algorithm
  - Let q denote the number of elements in a finite field. The number of points on an elliptic curve (denoted as #E) is bounded by
    $|\#E - (q+1)| \leq 2\sqrt{q}$ (Hasse's theorem)

Elliptic Curve Discrete Logarithm Problem

- In the elliptic curve group E, find a point G so that the order n of G is very large (if n is the same as the order of the group E, G is a generator of E)
  - The order (also called period) of an element b of a group is the smallest integer m so that $b^m = e$, where e is the identity element of the group
  - The order of an element G of an elliptic curve is the smallest integer n so that $n\text{G} = O$
    - Here nG means adding G to itself n times

Elliptic Curve Discrete Logarithm Problem

- For an elliptic curve group E over a finite field and an element G with order *n*, the discrete logarithm problem is stated as follows:

  Given a point Q which is a multiple of G, find the integer *x* so that *x*G = Q .

- Let the order of G be *n*, the above discrete logarithm problem requires about $\mathbf{2^{n/2}}$ operations to find the value *x*
  – Index Calculus Algorithm cannot be applied here

Elliptic Curve Discrete Logarithm Problem

- The elliptic curve group E over a finite field can be used to replace the group Z_p^* in those public key cryptosystems which are based on discrete logarithm problem
  - Then we obtain elliptic curve public key cryptosystems with much smaller key size

Elliptic Curve
Public Key Cryptosystem

Elliptic Curve Public Key Cryptosystem

- We will learn two elliptic curve public key cryptosystems
  – Elliptic curve Diffie-Hellman key exchange (ECDH)
  – Elliptic curve digital signature algorithm (ECDSA)

Elliptic Curve Public Key Cryptosystem

- In an elliptic curve public key cryptosystem, all the parties agree on the **domain parameters** which include:
  - The specification of the finite field
    - If GF(p) is used, then p defines the finite field (at least 256-bit)
    - If GF(2^m) is used, then m and the irreducible polynomial f defines the finite field (m is at least 256)
  - The specification of the elliptic curve: constants a, b
  - The generator G of a subgroup of E
  - The order n of G
  - The parameter h, where $h = \frac{\text{the order of the group E}}{\text{the order of the element G}}$, the value of h should be small ($h \leq 4$), and preferably, $h = 1$.

Elliptic Curve Diffie-Hellman key exchange

- In Diffie-Hellman key exchange, two system parameters: a large prime $\boldsymbol{p}$ and a generator $\boldsymbol{g}$ of the multiplicative cyclic group Z_p^*

	Alice	Bob
step 1:	generate random number r_a	generate random number r_b
step 2:	compute $Y_a = (g^{r_a}) \bmod p$	compute $Y_b = (g^{r_b}) \bmod p$
step 3:	send Y_a to Bob	send Y_b to Alice
step 4:	compute $K_a = (Y_b)^{r_a} \bmod p$	compute $K_b = (Y_a)^{r_b} \bmod p$

$$\boldsymbol{K_a = K_b}$$

Elliptic Curve Diffie-Hellman key exchange

- In elliptic curve Diffie-Hellman key exchange, all the parties use the same domain parameters

	Alice	Bob
step 1:	generate random integer r_a	generate random integer r_b
step 2:	compute $Y_a = r_a G$	compute $Y_b = r_b G$
step 3:	send Y_a to Bob	send Y_b to Alice
step 4:	compute $Q_a = r_a Y_b$	compute $Q_b = r_b Y_a \bmod p$

- After the exchange, $Q_a = Q_b = r_a r_b G$
  - The secret key is derived from the x coordinate of Q_a and Q_b

Elliptic Curve Digital Signature Algorithm

- In ElGamal digital signature, the key generation is:

1. Generate a large random prime p
2. Find a generator g of the multiplicative group Z_p^* .
3. Select a random secret integer x $(0 < a < p)$, and compute $y = g^x \bmod p$

Public key: (p, g, y)

Private key: x

Elliptic Curve Digital Signature Algorithm

- ElGamal signature generation
  - Choose a random secret k with $\gcd(k, p\text{-}1) = 1$
  - Compute $r = g^k \bmod \mathrm{p}$
  - Compute $s = (H(m)\text{-}xr)\, k^{-1} \pmod{p\text{-}1}$
  - If $s = 0$, start over again

  The signature of message m is: (r, s)
- ElGamal signature verification
  - $0 < r < p$, $0 < s < p\text{-}1$
  - $g^{H(\mathrm{m})} \bmod p \stackrel{?}{=} y^r r^s \bmod p$

Elliptic Curve Digital Signature Algorithm

- In ECDSA, all the parties use the same domain parameters
  - The order ***n*** of G is required to be a large **prime**
- The key generation of ECDSA is:

Select a random secret integer x $(0 < x < n)$, and compute $Y = xG$

Public key: Y

Private key: x

Elliptic Curve Digital Signature Algorithm

- ECDSA signature generation
  - Generate a one-time secret integer k less than n
  - Compute $(x_1, y_1) = k\mathrm{G}$
  - Compute $r = x_1 \bmod n$, $s = ((H(m)+xr)\, k^{-1}) \bmod n$
  - If $r = 0$ or $s = 0$, start over again

  The signature of the message m is: (r, s)

Elliptic Curve Digital Signature Algorithm

- ECDSA signature verification
  - Check that the public key Y is a valid point on the curve
  - Compute $w = s^{-1} \bmod n$
  - Compute $u_1 = \mathrm{H}(m)w \bmod n$, $u_2 = r\, w \bmod n$
  - Compute $(x_1, y_1) = u_1 G + u_2 Y$
  - If $r \equiv x_1 (\mathrm{mod}\ n)$ and $(x_1, y_1) \neq O$, the signature is valid

NIST Elliptic Curve Cryptosystem Standards

- NIST standardized two elliptic curve public key cryptosystems
  - Elliptic curve Diffie-Hellman key exchange (**ECDH**)
  - Elliptic curve digital signature algorithm (**ECDSA**)
- NIST standardized Dual_EC_DRBG (Dual Elliptic Curve Deterministic Random Bit Generator) that generates random numbers from a random seed
  - Withdrawn in 2015 due to the disclosure that NSA had inserted backdoor into this standard

NIST Elliptic Curve Cryptosystem Standards

- In the NIST standards ECDH and ECDSA, NIST recommended 15 elliptic curves together with the generators G for various security levels (FIPS 186-3)
  - 5 elliptic curves over GF(p)
  - 10 elliptic curves over GF(2^m)
- The company Certicom recommended 25 elliptic curves together with the generators G
  - 11 elliptic curves over GF(p)
  - 14 elliptic curves over GF(2^m)
- The NIST's curves are a subset of the Certicom curves
  - RFC 4492 Appendix A. Equivalent Curves

Elliptic Curve Examples

- The curve NIST P-256 is the same as Certicom's secp256r1 (256-bit prime)

```
p = FFFFFFFF 00000001 00000000 00000000
    00000000 FFFFFFFF FFFFFFFF FFFFFFFF
```

$= 2^{224}(2^{32}-1) + 2^{192} + 2^{96} - 1$

```
a = FFFFFFFF 00000001 00000000 00000000
    00000000 FFFFFFFF FFFFFFFF FFFFFFFC
b = 5AC635D8 AA3A93E7 B3EBBD55 769886BC
    651D06B0 CC53B0F6 3BCE3C3E 27D2604B
G =(6B17D1F2 E12C4247 F8BCE6E5 63A440F2
    77037D81 2DEB33A0 F4A13945 D898C296,
    4FE342E2 FE1A7F9B 8EE7EB4A 7C0F9E16
    2BCE3357 6B315ECE CBB64068 37BF51F5)
n = FFFFFFFF 00000000 FFFFFFFF FFFFFFFF
    BCE6FAAD A7179E84 F3B9CAC2 FC632551
h = 01
```

Elliptic Curve Examples

- Certicom's curve secp256k1 (256-bit prime) is used in ECDSA in many cryptocurrencies

```
p = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
    FFFFFFFF FFFFFFFF FFFFFFFE FFFFFC2F
```

$$= 2^{256} - 2^{32} - 2^{9} - 2^{8} - 2^{7} - 2^{6} - 2^{4} - 1$$

```
a = 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000000
b = 00000000 00000000 00000000 00000000
    00000000 00000000 00000000 00000007
G =(79BE667E F9DCBBAC 55A06295 CE870B07
    029BFCDB 2DCE28D9 59F2815B 16F81798,
    483ADA77 26A3C465 5DA4FBFC 0E1108A8
    FD17B448 A6855419 9C47D08F FB10D4B8)
n = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE
    BAAEDCE6 AF48A03B BFD25E8C D0364141
h = 01
```

Summary

- Elliptic curve over finite field
  - Group
  - Addition
  - Discrete logarithm problem is hard
- Elliptic curve public key cryptosystem
  - ECDH (NIST standard)
  - ECDSA (NIST standard)