Cryptography on Sage

Giulia Mauri

Politecnico di Milano

email: *giulia.mauri@polimi.it*
website: *http://home.deib.polimi.it/gmauri*

May 27, 2015

Overview

1. Elliptic Curve
   - Preliminary Concepts
2. Elliptic Curve Cryptography
   - ElGamal Cryptosystem
   - ElGamal Digital Signature
   - EC Diffie-Hellman Key Exchange
   - EC Digital Signature Algorithm

Definition

An Elliptic Curve E is the graph of an equation:

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_5$$

$a_1, a_2, a_3, a_4, a_5, x, y$ are defined in K where K is a field.
An elliptic curve is defined by:

`EllipticCurve(K, [a1, a2, a3, a4, a5])`
`EllipticCurve(K, [a4,a5])` *Weierstrass equation*

Definition

An Elliptic Curve E is the graph of an equation:

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_5$$

$a_1, a_2, a_3, a_4, a_5, x, y$ are defined in K where K is a field.
An elliptic curve is defined by:

`EllipticCurve(K, [a1, a2, a3, a4, a5])`
`EllipticCurve(K, [a4,a5])` *Weierstrass equation*

Example

```
sage: EllipticCurve(GF(11), [0,0,1,-1,0])
>> Elliptic Curve defined by y^2 + y = x^3 + 10*x
    over Finite Field of size 11
sage: EllipticCurve(Zmod(11), [1,1])
>> Elliptic Curve defined by y^2 = x^3 + x + 1
    over Ring of integers modulo 11
```

Addition

Exercise

Add the points A(1,3) and B(3,5) on the elliptic curve $y^2 = x^3 + 24x + 13 \pmod{29}$.

Addition

Exercise

Add the points A(1,3) and B(3,5) on the elliptic curve $y^2 = x^3 + 24x + 13 \pmod{29}$.

Solution

```
sage: E = EllipticCurve(Zmod(29),[24,13])
sage: A = E(1,3); B = E(3,5)
sage: C = A + B
sage: print C
>> (26 : 1 : 1)
sage: E.is_on_curve(26,1)
>> True
```

Addition

Exercise

Add the points A(1,3) and B(3,5) on the elliptic curve $y^2 = x^3 + 24x + 13 \pmod{29}$.

Solution

```
sage: E = EllipticCurve(Zmod(29),[24,13])
sage: A = E(1,3); B = E(3,5)
sage: C = A + B
sage: print C
>> (26 : 1 : 1)
sage: E.is_on_curve(26,1)
>> True
```

NOTE:

- `A = E(x,y)` defines a point on curve E
- `(X:Y:Z)` is a point in projective coordinates where (x,y)=(X/Z,Y/Z)
- `E.is_on_curve(x,y)` verifies if a point is on curve E

Points and Infinity Point

Exercise

Add A(1,3) to the point at infinity on the curve E. Then find out how many points has the curve and list all of them.

Points and Infinity Point

Exercise

Add A(1,3) to the point at infinity on the curve E. Then find out how many points has the curve and list all of them.

Solution

```
sage: E = EllipticCurve(Zmod(29),[24,13])
sage: A = E(1,3)
sage: D = E(0)
sage: F = A + D
sage: print F
>> (1 : 3 : 1)
sage: E.cardinality()
>> 38
sage: E.points()
>>[(0 : 1 : 0), (0 : 10 : 1), (0 : 19 : 1), (1 : 3 : 1),
(1 : 26 : 1), (3: 5 : 1), (3 : 24 : 1), (4 : 12 : 1), ...]
```

Points and Infinity Point

Solution (continue)

```
[...,  (4 : 17 : 1), (6 : 5 : 1), (6 : 24: 1), (9 : 1 : 1),
(9 : 28 : 1), (10 : 8 : 1), (10 : 21 : 1), (11 : 10 :1),
(11 : 19 : 1), (12 : 12 : 1), (12 : 17 : 1), (13 : 12 : 1),
(13 : 17: 1), (15 : 6 : 1), (15 : 23 : 1), (18 : 10 : 1),
(18 : 19 : 1), (19 : 7: 1), (19 : 22 : 1), (20 : 5 : 1),
(20 : 24 : 1), (21 : 11 : 1), (21 :18 : 1), (22 : 13 : 1),
(22 : 16 : 1), (23 : 1 : 1), (23 : 28 : 1),  (24: 0 : 1),
(26 : 1 : 1), (26 : 28 : 1)]
```

NOTE:

- `D = E(0)` defines the point at infinity
- `E.cardinality()` finds out how many are the points
- `E.points()` lists all the points

Order, Generator and Inverse

Exercise

Compute the order of the curve defined by $y^2 + y = x^3 + x^2 + x + 1$ *over the finite field with 701 elements, find a generator showing its order. Then compute the inverse of the point P(1,37).*

Order, Generator and Inverse

Exercise

Compute the order of the curve defined by $y^2 + y = x^3 + x^2 + x + 1$ *over the finite field with 701 elements, find a generator showing its order. Then compute the inverse of the point P(1,37).*

Solution

```
sage: E = EllipticCurve(GF(701), [0,1,1,1,1])
sage: E.order()        >> 711
sage: G = E.gen(0)     >> (165:29:1)
sage: G.order()        >> 711
sage: P = E(1,37)
sage: print -P         >> (1 : 663 : 1)
```

- `E.order()` gives the order of the curve
- `G.order()` gives the order of the point
- `-P` is the inverse of a point

Multiplication

Exercise

Let A(1,3) be a point on the elliptic curve $E : y^2 = x^3 + 24x + 13 \pmod{29}$. *Find 7A. Then find kA for k=1,2,..,40.*

Multiplication

Exercise

Let A(1,3) be a point on the elliptic curve $E : y^2 = x^3 + 24x + 13 \pmod{29}$*. Find 7A. Then find kA for k=1,2,..,40.*

Solution

```
sage: E = EllipticCurve(Zmod(29), [24,13])
sage: A = E(1,3)
sage: F = 7 * A
>> (15 : 6 : 1)
sage: for k in range(1,41):
...:            G = k * A
...:            print G
>> (1:3:1)   #k =1
>> (11:10:1) #k=2
>> ....
>> (0:1:0)   #k=19
```

Multiplication

Solution (continue)

```
>> ....
>> (0:1:0)       #k=38
>> (1:3:1)       #k=39
>> (11 : 10 : 1) #k=40
A.order()
>> 19
```

NOTE:

- `(0:1:0)` is the infinity point
- `A.order()` gives exactly 19

ElGamal Cryptosystem

Exercise

The elliptic curve E is $y^2 = x^3 + 3x + 45 \pmod{8831}$ *and the point is A(4,11). Alice's message is the point* $P_m(5, 1743)$*. Bob has chosen his secret random number* $a = 3$ *and has computed* $B = aA$*. Bob publishes this point b. Alice chooses the random number* $k = 8$ *and computes* $Y_1 = kA$ *and* $Y_2 = P_m + kB$*. Alice sends* Y_1, Y_2 *to Bob, who deciphers the message. Cipher and decipher the message.*

ElGamal Cryptosystem

Exercise

The elliptic curve E is $y^2 = x^3 + 3x + 45 \pmod{8831}$ and the point is A(4,11). Alice's message is the point $P_m(5, 1743)$. Bob has chosen his secret random number $a = 3$ and has computed $B = aA$. Bob publishes this point b. Alice chooses the random number $k = 8$ and computes $Y_1 = kA$ and $Y_2 = P_m + kB$. Alice sends Y_1, Y_2 to Bob, who deciphers the message. Cipher and decipher the message.

Solution

```
sage: E = EllipticCurve(Zmod(8831),[3,45])
sage: A = E(4,11); P_m = E(5,1743)
sage: a = 3; k = 8
sage: B = a * A
sage: Y_1 = k * A
sage: Y_2 = P_m + k * B
```

ElGamal Cryptosystem

Solution (continue)

```
sage: print B, Y_1, Y_2
>> (413:1808:1) (5415:6321:1) (6626:3576:1)
sage: m = Y_2 - Y_1 * a
>> (5:1743:1)
```

NOTE:

- (5:1743:1) is exactly the Alice's message P_m

ElGamal Cryptosystem

Exercise

Alice uses the public key ElGamal cryptosystem. She publishes the curve $E : y^2 \equiv x^3 + 2x + 2 \pmod{13}$ *and the point* $A = (3, 3)$ *of order 15. She also chooses a secret number* $a = 7$ *and publishes the point* $B = aA$*. Bob wants to send to Alice a message corresponding to the point* $P_m = (8, 6)$*.*
Questions:

1. *Calculate* B*.*
2. *Cipher the message using* $k = 3$*.*
3. *Decipher the message.*
4. *Using the repeated nonce, decipher the ciphered message* $(Y_{1,2} = (6, -3), Y_{2,2} = (2, 1))$

ElGamal Cryptosystem

Solution (1. $B = aA$)

```
sage: E = EllipticCurve(Zmod(13),[2,2])
sage: A = E(3,3); a = 7
sage: B = a * A
>>  (11 : 4 : 1)
```

Solution (2. $Y_1 = kA$ and $Y_2 = P_m + kB$)

```
sage: k = 3; P_m = E(8,6)
sage: Y_1 = k * A
>> (6 : 10 : 1)
sage: Y_2 = P_m  + k*B
>> (4 : 3 : 1)
```

Solution (3. $P_m = Y_2 - aY_1$)

```
sage: P = Y_2 - a * Y_1
>> (8 : 6 : 1)
```

Solution (4. $Y_{2,1} - P_{m,1} = kB = Y_{2,2} - P_{m,2}$)

```
sage: Y_22 = E(2,1)
sage: P_m2 = Y_22 - Y_2 + P_m
>> (6 : 10 : 1)
```

ElGamal Digital Signature

Exercise

Alice uses the following ElGamal signature with elliptic curves. Alice chooses the curve: $E : y^2 \equiv x^3 + 3 \pmod{31}$
The number $p = 31$ *is prime. Alice computes the number of points* n *which belong to the curve and obtain* $n = 43$*. On the curve* E *she chooses the point* $A = (1, 2)$ *and the secret number* $a = 18$*. She then computes the position of the point* $B = aA$ *and obtains* $B = aA = (17, 24)$*.*
Alice publishes the curve E*, the number* p *and the position of the points* A *and* B*. The number* a *is kept secret.*

1. *Alice wants to send the message* $m_1 = 7$ *and chooses the random number* $k = 3$*. Compute Alice's signature.*
2. *Verify the signature.*
3. *Alice, then, signs a second message* $m_2 = 13$ *and uses the same nonce as before, obtaining* $R_2 = (22, 24), s_2 = 30$*. Bob computes the nonce.*

ElGamal Digital Signature

Solution (1. $R = kA$ and $s = k^{-1}(m_1 - ax_R)$)

```
sage: E = EllipticCurve(Zmod(31),[0,3])
sage: A = E(1,2); a = 18; k = 3; m_1 = 7; n = 43
sage: B = a*A
>> (17 : 24 : 1)
sage: R = k*A
>> (22 : 24 : 1)
sage: (x,y) = R
sage: kinv = int(mod(k^(-1),n))
>> 29
sage: c = mod(m_1 - (a*int(x)),n)
sage: s_1 = mod(kinv*c, n)
>> 28
```

ElGamal Digital Signature

Solution (2. $V_1 = x_R B + sR$ and $V_2 = mA$)

```
sage: V_1 = int(x)*B + int(s)*R
>> (25 : 29 : 1)
sage: V_2 = m_1 * A
>> (25 : 29 : 1)
```

Solution (3. $s_1 k - m_1 = -a x_R = s_2 k - m_2$)

```
sage: m_2 = 13; s_2 = 30
sage: s = mod((s_1 - s_2),n)
>> 41
sage: m = mod((m_1 - m_2),n)
>> 37
sage: sinv = mod(41^(-1),n)
>> 21
sage: k = sinv * m
>> 3
```

ECDHKE

Exercise

The elliptic curve E is $y^2 = x^3 + x + 7206 \pmod{7211}$ *and the point is G(3,5). Alice chooses her secret* $N_A = 12$ *and Bob chooses his secret* $N_B = 23$. *Simulate the DH key exchange.*

ECDHKE

Exercise

The elliptic curve E is $y^2 = x^3 + x + 7206 \pmod{7211}$ *and the point is G(3,5). Alice chooses her secret* $N_A = 12$ *and Bob chooses his secret* $N_B = 23$*. Simulate the DH key exchange.*

Solution (0. Setup parameters)

```
sage: E = EllipticCurve(Zmod(7211),[1,7206])
sage: G = E(3,5)
sage: N_A = 12
sage: N_B = 23
```

ECDHKE

Exercise

The elliptic curve E is $y^2 = x^3 + x + 7206 \pmod{7211}$ *and the point is G(3,5). Alice chooses her secret* $N_A = 12$ *and Bob chooses his secret* $N_B = 23$*. Simulate the DH key exchange.*

Solution (0. Setup parameters)

```
sage: E = EllipticCurve(Zmod(7211),[1,7206])
sage: G = E(3,5)
sage: N_A = 12
sage: N_B = 23
```

Solution (1. Alice calculates A and sends it to Bob)

```
sage: A = N_A * G
>> (1794 : 6375 : 1)
```

ECDHKE

Solution (2. Bob calculates B and sends it to Alice)

```
sage: B = N_B * G
>> (3861 : 1242 : 1)
```

ECDHKE

Solution (2. Bob calculates B and sends it to Alice)

```
sage: B = N_B * G
>> (3861 : 1242 : 1)
```

Solution (3. Alice takes B and multiplies by N_A to get the key)

```
sage: K_A = B * N_A
>> (1472 : 2098 : 1)
```

Solution (2. Bob calculates B and sends it to Alice)

```
sage: B = N_B * G
>> (3861 : 1242 : 1)
```

Solution (3. Alice takes B and multiplies by N_A to get the key)

```
sage: K_A = B * N_A
>> (1472 : 2098 : 1)
```

Solution (4. Bob takes A and multiplies by N_B to get the key)

```
sage: K_B = A * N_B
>> (1472 : 2098 : 1)
```

Note that they must have the same key.

Functions for ECDHKE

Exercise

Assume that your domain parameters are: Elliptic Curve defined by $y^2 = x^3 + 26484x + 15456$ *over Finite Field of size 63709,* $q = 63839$, $G = (53819,6786)$.

1. *Write a function that takes a curve, and a base point on the curve and generates the secret value* x *and the public value* X *as per ECDH.*
2. *Write a function that takes a public value and a secret value and computes the shared secret.*
3. *Show your functions work by simulating an ECDH key exchange.*

Functions for ECDHKE

Solution (0. Setup parameters)

```
sage: E = EllipticCurve(GF(63709), [26484,15456])
sage: G = E(53819,6786)
sage: q = G.order()
```

Functions for ECDHKE

Solution (0. Setup parameters)

```
sage: E = EllipticCurve(GF(63709), [26484,15456])
sage: G = E(53819,6786)
sage: q = G.order()
```

Solution (1. def function1(G,q))

```
sage: def function1(G,q):
...:          x = random.randint(2,q-1)
...:          X = x * G
...:         return x, X
```

Functions for ECDHKE

Solution (0. Setup parameters)

```
sage: E = EllipticCurve(GF(63709), [26484,15456])
sage: G = E(53819,6786)
sage: q = G.order()
```

Solution (1. def function1(G,q))

```
sage: def function1(G,q):
...:           x = random.randint(2,q-1)
...:           X = x * G
...:          return x, X
```

Solution (2. def function2(X,x))

```
sage: def function2(X,x):
...:           sharedsecret = X * x
...:          return sharedsecret
```

Functions for ECDHKE

Solution (3. Key Exchange)

```
sage: (a,A) = function1(G,q)      #executed by Alice
sage: (b,B) = function1(G,q)      #executed by Bob
sage: sharedsecretA = function2(B,a) #executed by Alice
sage: sharedsecretB = function2(A,b) #executed by Bob
sage: if sharedsecretA == sharedsecretB:
...:          print sharedsecretA
...:    else:
...:           print -1
>> ( 10484 : 24536 : 1 )
```

ECDSA

Exercise

Alice uses the DSA signature scheme on the elliptic curve $E : y^2 \equiv x^3 + 2x + 6 \mod 7$. *The curve* E *has* 11 *points. Alice chooses the base point* $A = (1, 3)$, *the secret* $a = 4$ *and computes* $B = aA$. *Then she signs the message* $m_1 = 3$ *using the nonce* $k = 6$.

1. Compute B.

2. Sign m_1.

3. Verify the signature obtained in 2.

ECDSA

Exercise

Alice uses the DSA signature scheme on the elliptic curve $E : y^2 \equiv x^3 + 2x + 6 \mod 7$. *The curve* E *has* 11 *points. Alice chooses the base point* $A = (1, 3)$, *the secret* $a = 4$ *and computes* $B = aA$. *Then she signs the message* $m_1 = 3$ *using the nonce* $k = 6$.

1. Compute B.

2. Sign m_1.

3. Verify the signature obtained in 2.

Solution (1. $B = aA$)

```
sage: p=7; k = 6; m = 3; n = 11
sage: E = EllipticCurve(Zmod(p),[2,6])
sage: A = E(1,3), a = 4
sage: B = A * a
>> (3 : 5 : 1)
```

ECDSA

Solution (2. $R = kA$ and $s = k^{-1}(m + ax_R)$)

```
sage: R = k * A
>> (4 : 6 : 1)
sage: (x,y,z) = R
sage: kinv = int(mod(k^(-1),n))
>> 2
sage: c = mod(m + (a*int(x)),n)
sage: s = mod((kinv*c),n)
>> 5
```

ECDSA

Solution (2. $R = kA$ and $s = k^{-1}(m + ax_R)$)

```
sage: R = k * A
>> (4 : 6 : 1)
sage: (x,y,z) = R
sage: kinv = int(mod(k^(-1),n))
>> 2
sage: c = mod(m + (a*int(x)),n)
sage: s = mod((kinv*c),n)
>> 5
```

Solution (3. $u_1 = s^{-1}m$, $u_2 = s^{-1}x_R$ and $V = u_1A + u_2B$)

```
sage: u_1 = mod(m * s^(-1), n)                 >> 5
sage: u_2 = mod(int(x) * s^(-1), n)            >> 3
sage: V = int(u_1) * A + int(u_2) * B
>> (4 : 6 : 1)
```

NOTE: since V=R, the signature is verified.

ECDSA

Exercise

Consider the elliptic curve $E : y^2 \equiv x^3 + 3 \pmod 7$, *with 13 points. Alice publishes the curve and the points* $A = (1, 2)$ *and* $B = aA = (2, 2)$. *Then, Alice signs the messages* $m_1 = 2$ *and* $m_2 = 3$ *using the DSA signature and obtains:*

$$sig(m_1, k_1) = (x_{R,1}, s_1) = (3, 10)$$

$$sig(m_2, k_2) = (x_{R,2}, s_2) = (3, 5)$$

1. *List all the points of curve* E
2. *Verify the signature of message* m_1
3. *Using the repeated nonce, compute* k_1
4. *Compute the secret number* a

ECDSA

Solution (1. E.points())

```
sage: E = EllipticCurve(Zmod(7), [0,3])
sage: A = E(1,2); B = E(2,2); n = 13
sage: E.points()
>> [(0 : 1 : 0), (1 : 2 : 1), (1 : 5 : 1), (2 : 2 : 1),
(2 : 5 : 1), (3 : 3 : 1), (3 : 4 : 1), (4 : 2 : 1),
(4 : 5 : 1), (5 : 3 : 1), (5 : 4 : 1), (6 : 3 : 1),
(6 : 4 : 1)]
```

Solution (2. $u_1 = s^{-1}m$, $u_2 = s^{-1}x_R$ and $V = u_1A + u_2B$)

```
sage: m_1 = 2; x_R1 = 3; s_1 =10;
sage: u_1 = mod(s_1^(-1)*m_1,n)          >> 8
sage: u_2 = mod(s_1^(-1)*x_R1,n)         >> 12
sage: V = int(u_1)*A + int(u_2)*B
>>  (3 : 3 : 1)
```

ECDSA

Solution (3. $s_1 k - m_1 = a x_r = s_2 k - m_2$)

```
sage: m_2 = 3; x_R2 = 3; s_2 =5
sage: s = mod(s_1 - s_2,n)
>> 5
sage: m = mod(m_1 - m_2,n)
>> 12
sage: k = mod(m*s^(-1),n)
>> 5
```

Solution (4. $s_1 k - m_1 = a x_R$)

```
sage: a = mod((s_1*int(k) - m_1)*x_R1^(-1),n)
>> 3
```

ECDSA

Exercise

Alice uses the DSA signature scheme on the elliptic curve $E : y^2 \equiv x^3 + 2x + 6 \pmod 7$, *with 11 points. Alice chooses the base point* $A = (1, 3)$, *the secret* $a = 4$ *and computes* $B = aA$. *Then, Alice signs the message* $m_1 = 3$ *using the nonce* $k = 6$.

1. *Verify whether A satisfies the conditions required by DSA signature.*
2. *Compute B*
3. *Sign* m_1
4. *Verify the signature obtained in 3.*
5. *Alice signs the message* $m_2 = 4$ *and publishes* $sig(m_2) = [R_2, s_2, m_2] = [(4, 6), 7, 4]$. *Which mistake did she do? How can it be exploited by an attacker to find the secret key a?*

ECDSA

Solution (1. `A.order()`)

```
sage: E = EllipticCurve(Zmod(7), [2,6])
sage: A = E(1,3); a = 4; m_1 = 3; k = 6; n = 11
sage: A.order()
>> 11
```

Solution (2. $B = aA$)

```
sage: B = a*A
>> (3 : 5 : 1)
```

Solution (3. $R = kA$ and $s = k^{-1}(m + ax_R)$)

```
sage: R = k*A
>> (4 : 6 : 1)
sage: (x,y,z) = R
sage: s_1 = mod(k^(-1)*(m_1 + a*int(x)),n)
>> 5
```

ECDSA

Solution (4. $u_1 = s^{-1}m$, $u_2 = s^{-1}x_R$ and $V = u_1A + u_2B$)

```
sage: u_1 = mod(s_1^(-1)*m_1,n)
>> 5
sage: u_2 = mod(s_1^(-1)*int(x), n)
>> 3
sage: V = int(u_1)*A + int(u_2)*B
>> (4 : 6 : 1)
```

Solution (5. $s_1k - m_1 = ax_r = s_2k - m_2$ and $s_1k - m_1 = ax_R$)

```
sage: m_2 = 4; s_2 = 7
sage: s = mod(s_1 - s_2, n)          >> 9
sage: m = mod(m_1 - m_2, n)          >> 10
sage: k = mod(m*s^(-1),n)
>> 6
sage: xinv = mod(int(x)^(-1),n)
sage: a = mod(xinv*(s_1*k-m_1),n)
```