Some exercises on isogeny-based cryptography

Post-Quantum Crypto Mini-School 2022, Taipei, Taiwan

Lorenz Panny, Academia Sinica

July 14, 2022

1 Warm-up: ECDH

Define the elliptic curve

$$E\colon y^2 = x^3 + 486662x^2 + x$$

over the field $\mathbb{F}_p$ where

$$p = 2^{255} - 19\,.$$

This curve is widely known as *Curve25519*.

1. Find a $y \in \mathbb{F}_p$ such that the point $G = (9, y)$ lies on E.
2. Construct the curve E and the point G in Sage.
3. Verify that the order of G is a large prime.
4. Choose two secret keys $a, b \in \mathbb{Z}$ for Alice and Bob and check that $[b]([a]G) = [a]([b]G)$.
   (Recall that Sage simply uses the $\star$ operator for elliptic-curve scalar multiplication.)

2 Implement toy SIDH

In this exercise we shall play around with a miniaturized SIDH using the following parameters:

$$\begin{aligned}
p &= 2^{33} \cdot 3^{19} - 1 \\
E\colon\quad & y^2 = x^3 + x \;/\mathbb{F}_{p^2} \\
P &= (5510597594428399451, 5325375497072273612) \\
Q &= (1191331958183761542, 67757071284813610\mathbf{43i}) \\
R &= (3269987395986306519, 991123227256711357) \\
S &= (494348537840729533, 36457725058259493\mathbf{25i})
\end{aligned}$$

where $\mathbf{i}$ is a *fixed* square root of -1 in $\mathbb{F}_{p^2}$. (Careful: `GF(p^2)(-1).sqrt()` returns a random square root each time!)

You are Alice. Suppose your own secret key is $a = 123456789$. Using the description from the slides, compute the public key $(E/A, \varphi_A(R), \varphi_B(S))$.

Suppose further that you've received Bob's public key:

$$\begin{aligned}
E/B\colon\quad & y^2 = x^3 + (3006446413444749267 + 19669959555500013\mathbf{73i})x^2 + x \\
\varphi_B(P) &= (9376556621701166759 + 67405436466614889\mathbf{15i}, \\
&\qquad 1448317503955695488 + 90275868736750272\mathbf{99i}) \\
\varphi_B(Q) &= (9294696178641241049 + 80536795390027344\mathbf{03i}, \\
&\qquad 7475161061376416393 + 61017860534506788\mathbf{80i})
\end{aligned}$$

Compute the shared secret, i.e., the j-invariant of the elliptic curve $(E/B)/A'$.

Correct result: $1222534494549305449 + 1205746670324565\mathbf{49i}$.

3 Implement toy CSIDH

In this exercise, we compute CSIDH steps for $p = 38798759$, using $E_0 \colon y^2 = x^3 + x$ over $\mathbb{F}_p$ as the starting curve. Note that the ℓ_i in this case are $3, 5, 7, 11, 13, 17, 19$.

- Write a function that takes an elliptic curve $E/\mathbb{F}_p$ of order $p + 1$ and one of the ℓ_i, and returns a random point on E of order ℓ_i. (See [1, slide 50]; we are only doing "left" steps here for simplicity.)
- Use this function to compute a sequence of various ℓ_i-isogeny steps, and convince yourself that the resulting curve is independent of the order you are taking the steps in.

(Note: Technically, the theory only guarantees curves *isomorphic* over $\mathbb{F}_p$. However, unless you do something unnecessarily complicated, Sage will actually give you the exact *same* curves, which is due to the specific isogeny formulas used in Sage. Keep in mind that this is not always true.)

4 Break "BAD SIDH"

This exercise is about destroying the "simplified" SIDH variant shown on my last slide [1], with the following parameters:

$$\begin{aligned} p &= 2^{61} - 1 \\ E\colon &\quad y^2 = x^3 + x \ /\mathbb{F}_{p^2} \\ P &= (6, 320792492009892693) \\ Q &= (4, 7350864349621616 64\mathbf{i}) \end{aligned}$$

As a first experiment, repeatedly pick some random secrets for Alice and Bob and see which shared secret they would obtain.[1] Can you see why the protocol is broken? Can you explain why this happens?

The weakness extends even further: You can efficiently recover secret keys! So, suppose Bob's public key is as follows:

$$\begin{aligned} E/B\colon &\quad y^2 = x^3 + (2214526821652541414 + 1129714413825706394\mathbf{i})x^2 + x \\ \varphi_B(P) &= (2228932407989380877 + 18748354419847500 55\mathbf{i}, \\ &\qquad 22639666003584723 77 + 10643518028687653 91\mathbf{i}) \\ \varphi_B(Q) &= (238932792440849742 + 8498307101882743 02\mathbf{i}, \\ &\qquad 18545632180056807 21 + 21018459916358973 89\mathbf{i}) \end{aligned}$$

Find Bob's secret b.

References

[1] Lorenz Panny. *Isogeny-based Cryptography* I & II. Presentation slides. 14th July 2022. URL: `https://yx7.cc/docs/isog/isog_taipei_slides.pdf`.

[1] Since you have both secret keys here, you can skip most of the SIDH protocol and directly compute $E/\langle A, B\rangle$ by simply passing a *list* containing both kernel points to `E.isogeny()`.

Sample solutions

1 Warm-up: ECDH

```
sage: # 1.
sage: p = 2^255-19
sage: R.<X> = GF(p)[]
sage: f = X^3 + 486662*X^2 + X  # curve right-hand side
sage: y = Mod(f(9), p).sqrt()
sage: y
14781619447589544791020593568409986887264606134616475288964881837755586237401
sage:
sage: # 2.
sage: E = EllipticCurve(GF(p), [0,486662,0,1,0])
sage: G = E(9, y)
sage: G
(9 : 14781619447589544791020593568409986887264606134616475288964881837755586237401 : 1)
sage:
sage: # 3.
sage: G.order().factor()
72370055773322622139731865630429942408571163593799076060019509382854542509 89
sage:
sage: # 4.
sage: a = randrange(G.order())
sage: b = randrange(G.order())
sage: b*(a*G) == a*(b*G) == (a*b)*G
True
```

For finding y given x, there is actually a built-in method in Sage: `E.lift_x(9)`.

2 Implement toy SIDH

First, we convert all the given data to Sage:

```
sage: # parameters
sage: p = 2^33 * 3^19 - 1
sage: R.<X> = GF(p)[]
sage: F.<i> = GF(p^2, modulus=X^2+1)
sage: i^2 == -1
True
sage:
sage: E = EllipticCurve(F, [1,0])
sage: E
Elliptic Curve defined by y^2 = x^3 + x over Finite Field in i of size 9983749980331966463^2
sage: P = E(5510597594428399451, 5325375497072273612)
sage: Q = E(1191331958183761542, 6775707128481361043*i)
sage: R = E(3269987395986306519, 991123227256711357)
sage: S = E(494348537840729533, 3645772505825949325*i)
sage: [pt.order().factor() for pt in (P,Q,R,S)]
[2^33, 2^33, 3^19, 3^19]
sage:
sage: # given public key
sage: EB = EllipticCurve(F, [0, 3006446413444749267 + 1966995955550001373*i, 0, 1, 0])
sage: phiB_P = EB(9376556621701166759 + 6740543646661488915*i, 1448317503955695488 +
                                                    9027586873675027299*i)
sage: phiB_Q = EB(9294696178641241049 + 8053679539002734403*i, 7475161061376416393 +
                                                    6101786053450678880*i)
sage:
sage: # our own secret key
sage: a = 123456789
```

Alice's public key:

```
sage: A = P + a*Q
sage: phiA = E.isogeny(A, algorithm="factored")
sage: EA = phiA.codomain()
sage: phiA_R = phiA(R)
sage: phiA_S = phiA(S)
sage: print("Alice's public key:", (EA, phiA_R, phiA_S))
Alice's public key:
    (Elliptic Curve defined by
        y^2 = x^3 + (3754683794202648791*i+4442154660288608516)*x
                  + (5775103125841378254*i+2817405823585305307)
        over Finite Field in i of size 9983749980331966463^2,
    (1928436010540933667*i + 7177454073043552212
        : 7850572149588423574*i + 5331787871643853307 : 1),
    (7597229663287320697*i + 5362455923037443444
        : 2842356052343942415*i + 3159344315577691160 : 1))
```

The shared secret:

```
sage: phiB_A = phiB_P + a*phiB_Q
sage: psiA = EB.isogeny(phiB_A, algorithm="factored")
sage: EAB = psiA.codomain()
sage: print("shared secret:", EAB.j_invariant())
shared secret: 1205746670324456549*i + 1222534494549305449
```

3 Implement toy CSIDH

```
sage: p = 38798759
sage: E0 = EllipticCurve(GF(p), [1,0])
sage: factor(p+1)
2^3 * 3 * 5 * 7 * 11 * 13 * 17 * 19
sage: ls = [3,5,7,11,13,17,19]
sage:
sage: # 1.
sage: def point_of_order(E, l):
sage:     assert l in ls
sage:     while True:
sage:         P = E.random_point()
sage:         P *= (p+1)//l
sage:         if P:
sage:             return P
sage:
sage: point_of_order(E0, 7)
(7438993 : 2308968 : 1)
sage:
# 2.
sage: def step(E, l):
sage:     K = point_of_order(E, l)
sage:     phi = E.isogeny(K)
sage:     assert phi.degree() == l
sage:     return phi.codomain()
sage:
sage: step(step(E0, 5), 17)
Elliptic Curve defined by y^2 = x^3 + 1603945*x + 9405157 over Finite Field of size 38798759
sage: step(step(E0, 17), 5)
Elliptic Curve defined by y^2 = x^3 + 1603945*x + 9405157 over Finite Field of size 38798759
```

4 Break "BAD SIDH"

Left open as a challenge. Feel free to email me about this one. :-)