SIGPwny

General FA2026 • 2026-11-08

Cryptography II

Ahmed Alkhalawi

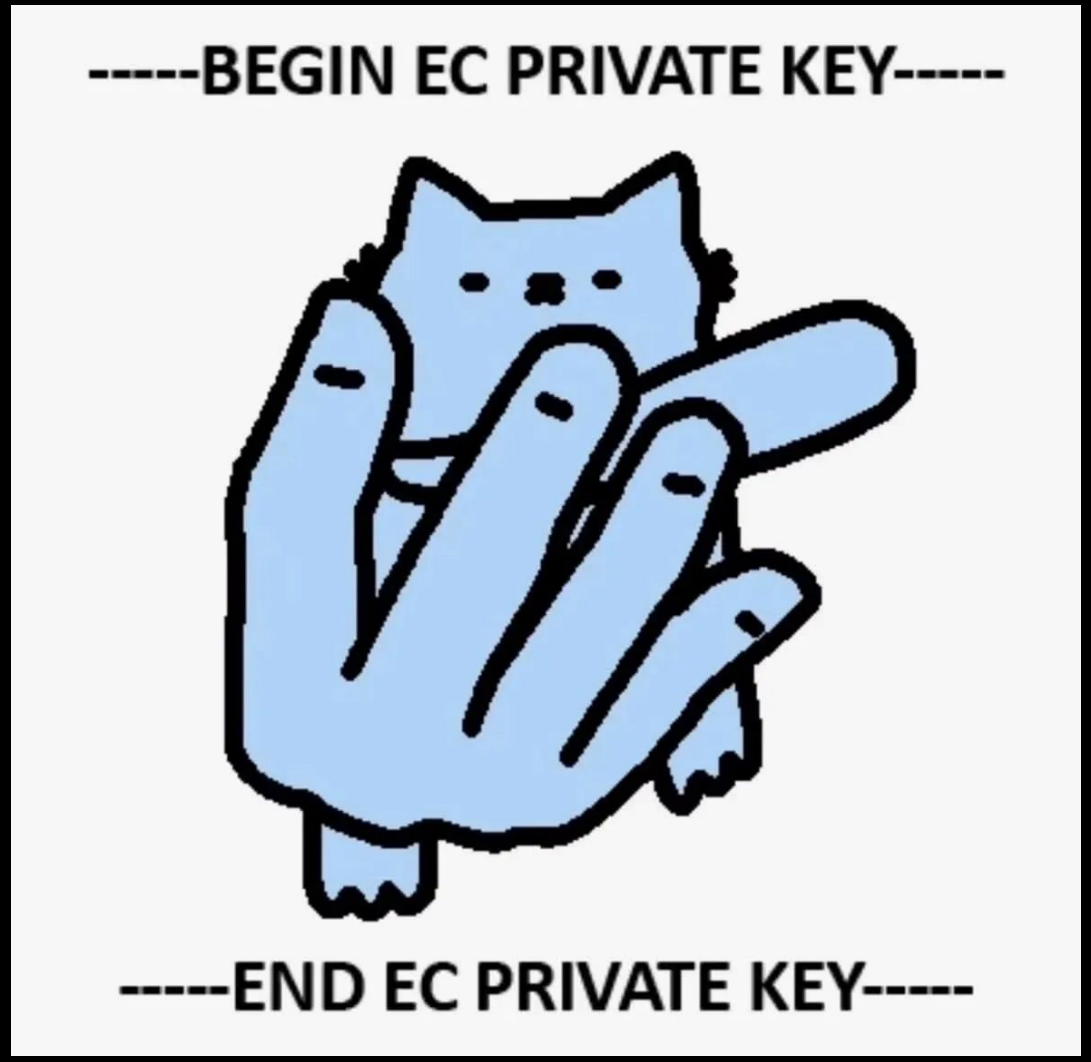
Announcements

- We will be playing **osu!gaming ctf** tomorrow!
  - Likely Siebel CS 2406 as usual, may change based on availability!



ctf.sigpwny.com

sigpwny{4}



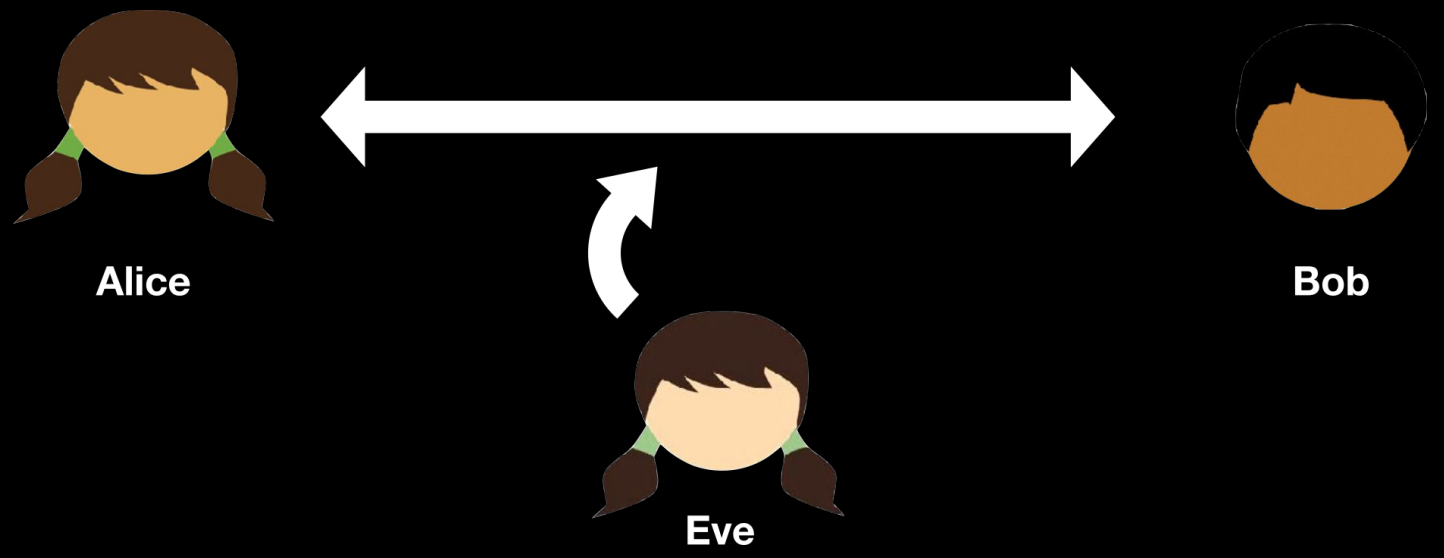
Contents

- What is Asymmetric Cryptography?
- Direct Encryption, RSA
- Key Exchange, (EC)DH
- Post Quantum Cryptography
  - Lattices
  - Isogenies



Asymmetric Cryptography

Good when Alice and Bob have no prior information

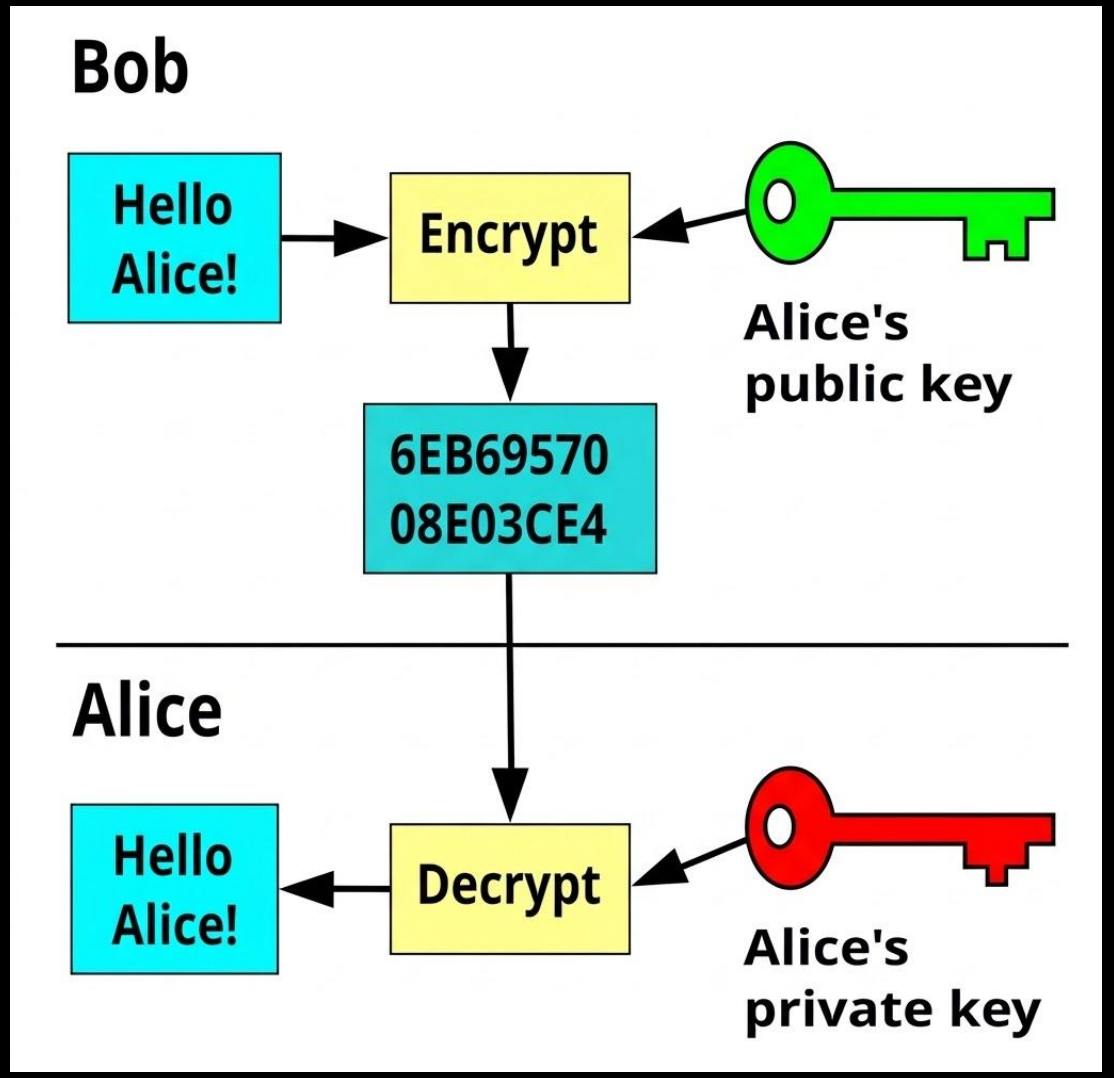


Asymmetric Encryption

- Public key
  - Intentionally broadcast for people to use
  - Anyone can use to encrypt a message to send us
- Private key
  - Keep to yourself
  - Use to decrypt other people's messages to us

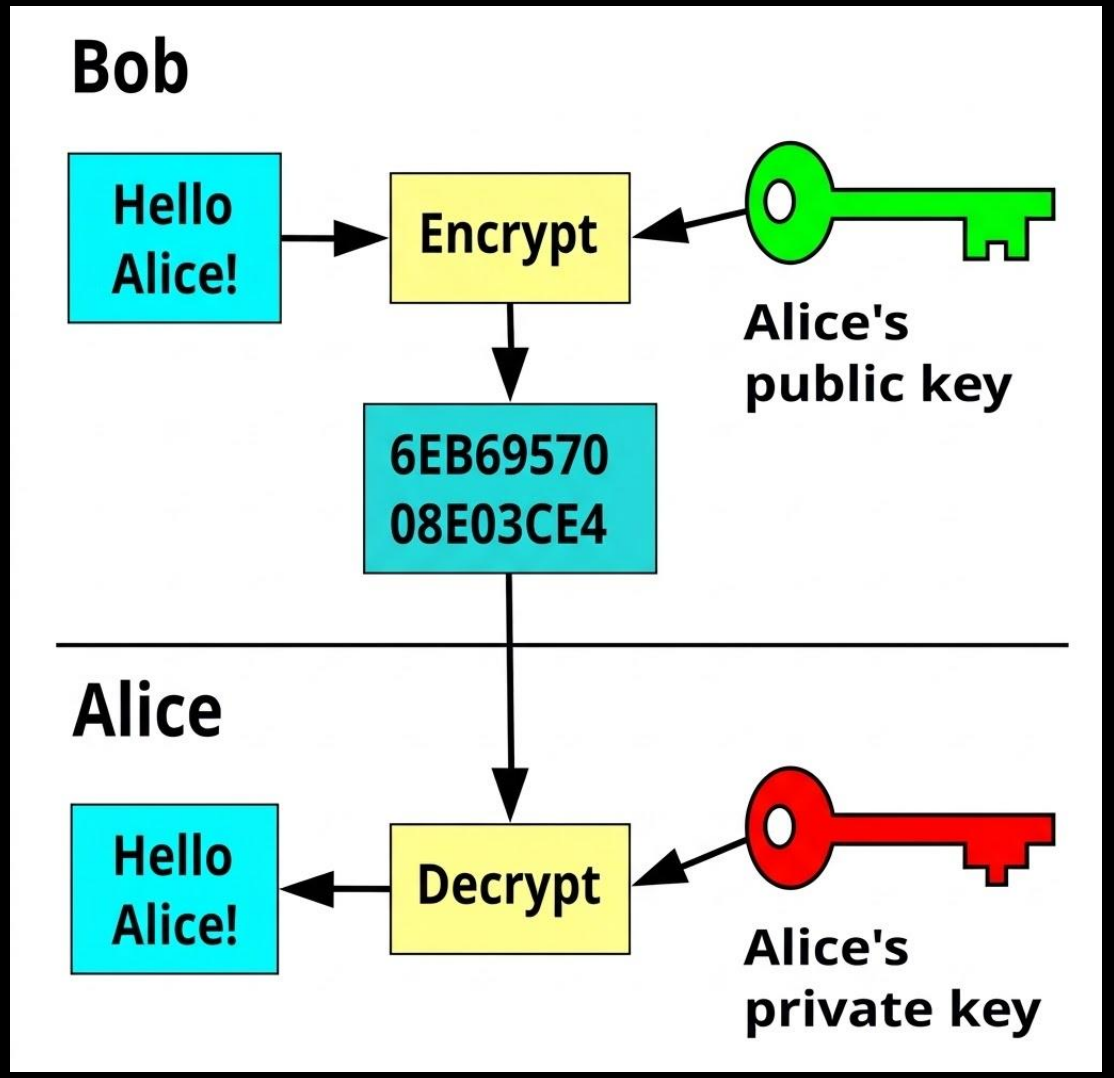


Direct Encryption



Direct Encryption

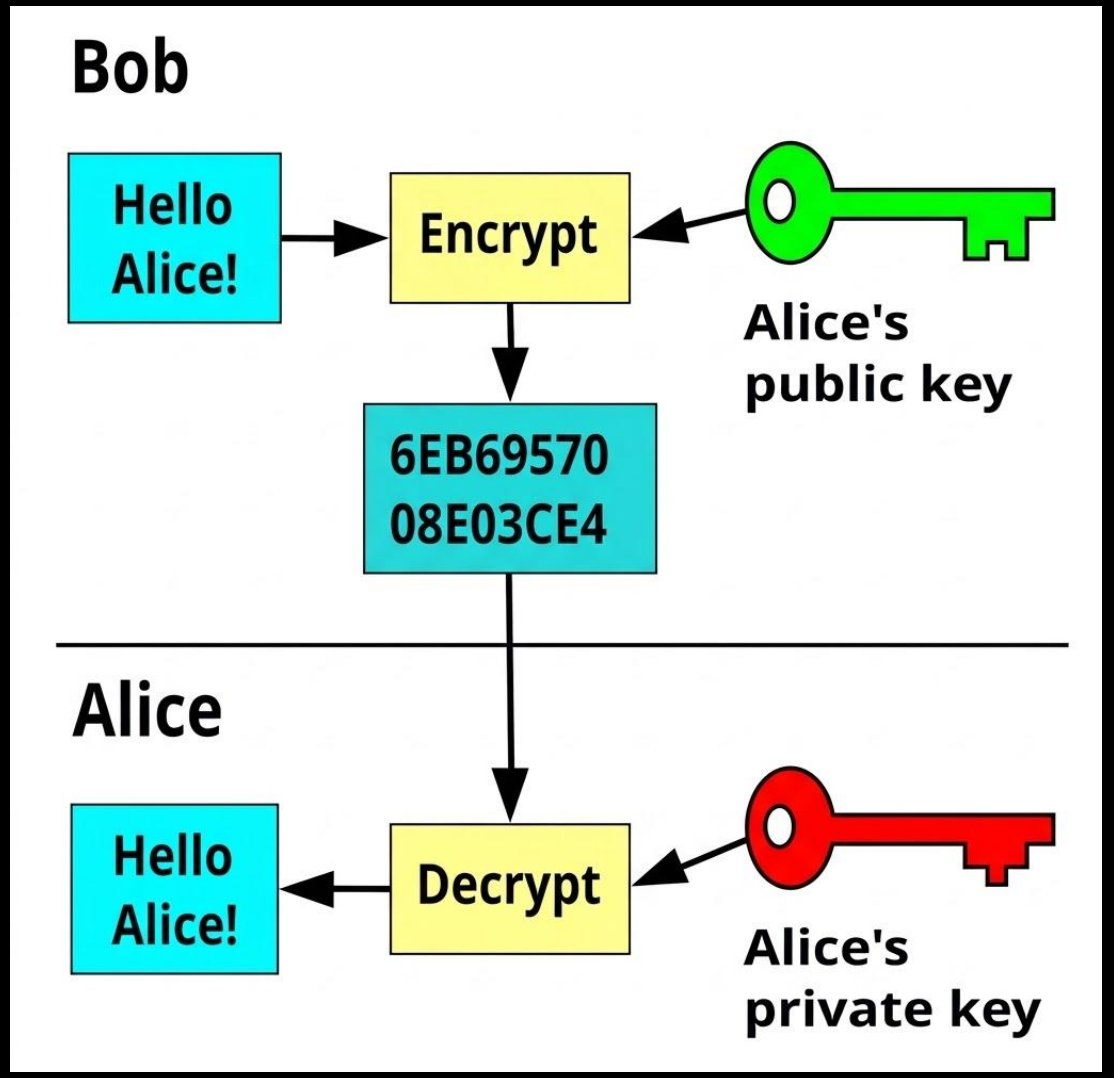
This method relies on a *key generation function* giving pk, sk such that its hard to compute sk from pk



Direct Encryption

And a *trapdoor function*
enc
Computing c = enc(m, pk) should be easy, but reversing it should be hard without sk

We should also have a decryption function dec(c, sk) = m



RSA

One example of this is RSA

First Alice generates random large (1024 bit) primes p, q as her private key, and releases their product n = p*q as her public key.

To encrypt a message m, Bob computes $c = m^e \bmod n$ where $e = 2^{16} + 1 = 65537$



RSA

To decrypt, we use Euler's Theorem: if we know d such that $e*d = 1 \mod (p-1)*(q-1)$, then $m^{e*d} = m \mod (p*q)$

Since Alice knows $(p-1)*(q-1)$, she can compute d in polynomial time then decrypt: $c^d = (m^e)^d = m \mod n$

However, as an eavesdropper doesn't know p and q, there's no known fast way to compute $(p-1)*(q-1)$ or to decrypt the message



RSA Attacks

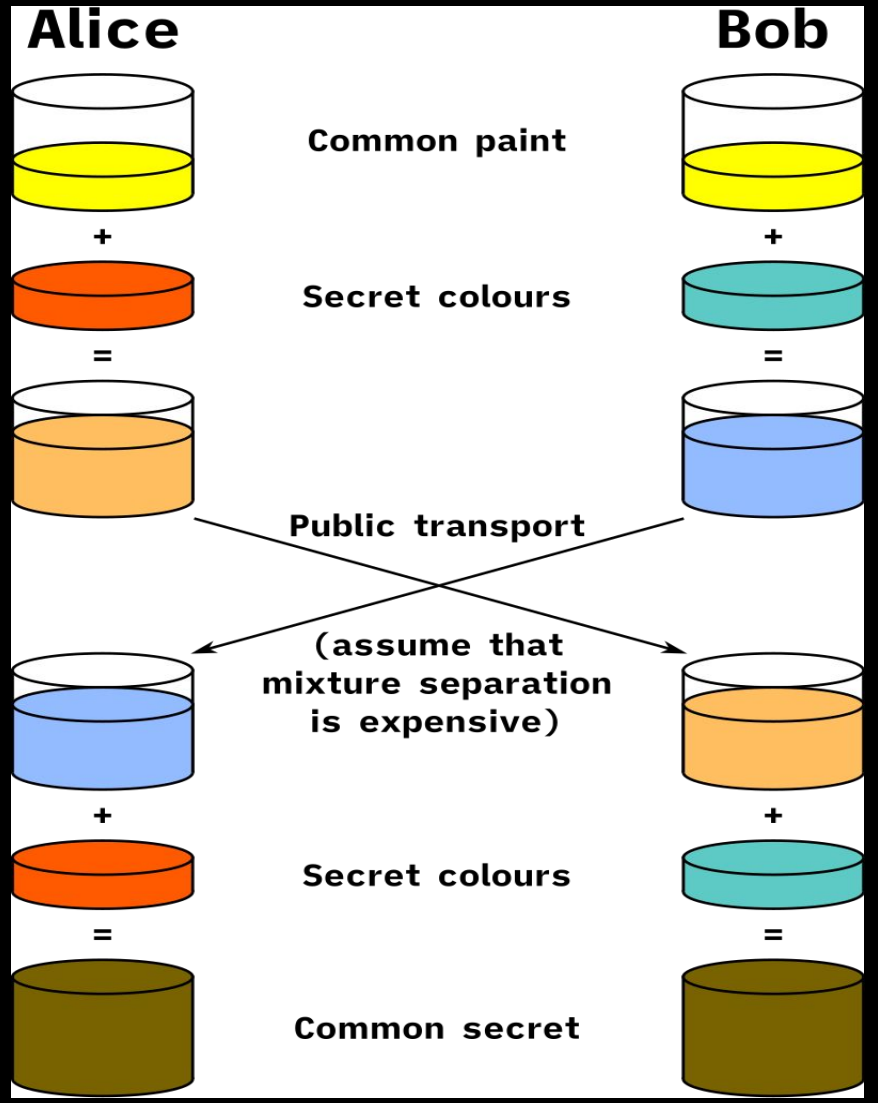
Common mistakes in RSA:

- Bad RNG when choosing prime (ROCA, 750000 Estonian IDs)
- Using small e (Coppersmith's Attack)
- Using small d (Wiener's attack)
- Padding Oracles

- Stop Using RSA
- Practice on CryptoHack!

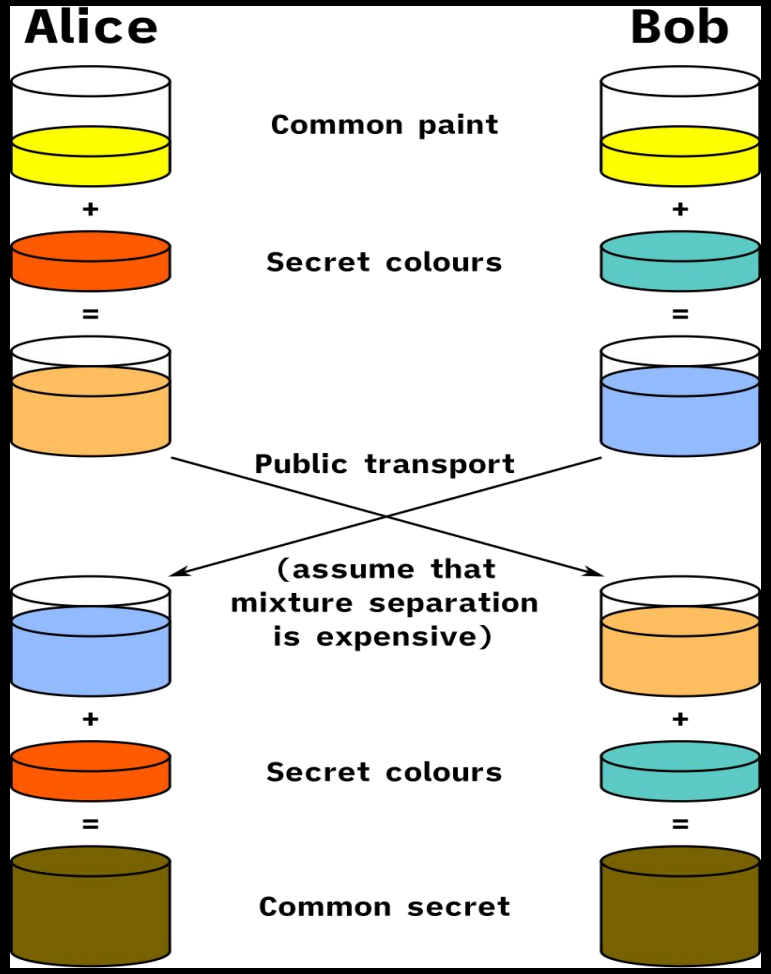


Key Exchange



Key Exchange

This method relies on a key-generation function as well as a one-way function f such that
f(pk_A, sk_B) = f(pk_B, sk_A)



Diffie Hellman Key Exchange

One example is Diffie Hellman.

First choose a random 2048 bit prime p.

First, Alice and Bob both generate random numbers a, b as private keys

Then compute $A = 2^a \bmod p$, $B = 2^b \bmod p$ as public keys and share them

Then the shared secret is $A^b = B^a = 2^{a*b} \bmod p$

Now just use AES!



DH Attacks

- Not using random p (Logjam)
- If p-1 is *smooth,* ie it factorizes into a product of many small primes, then you can use Pohlig-Hellman
  - Usually, we use a safe prime p = 2q+1



DH Attacks

- Not using random p (Logjam)
- If p-1 is *smooth,* ie it factorizes into a product of many small primes, then you can use Pohlig-Hellman
  - Usually, we use a safe prime p = 2q+1

As we can see, there are far fewer attacks than RSA.

However, we still have to generate a prime and private keys are large due to subexponential attacks, can we do better?



Elliptic Curve Diffie Hellman

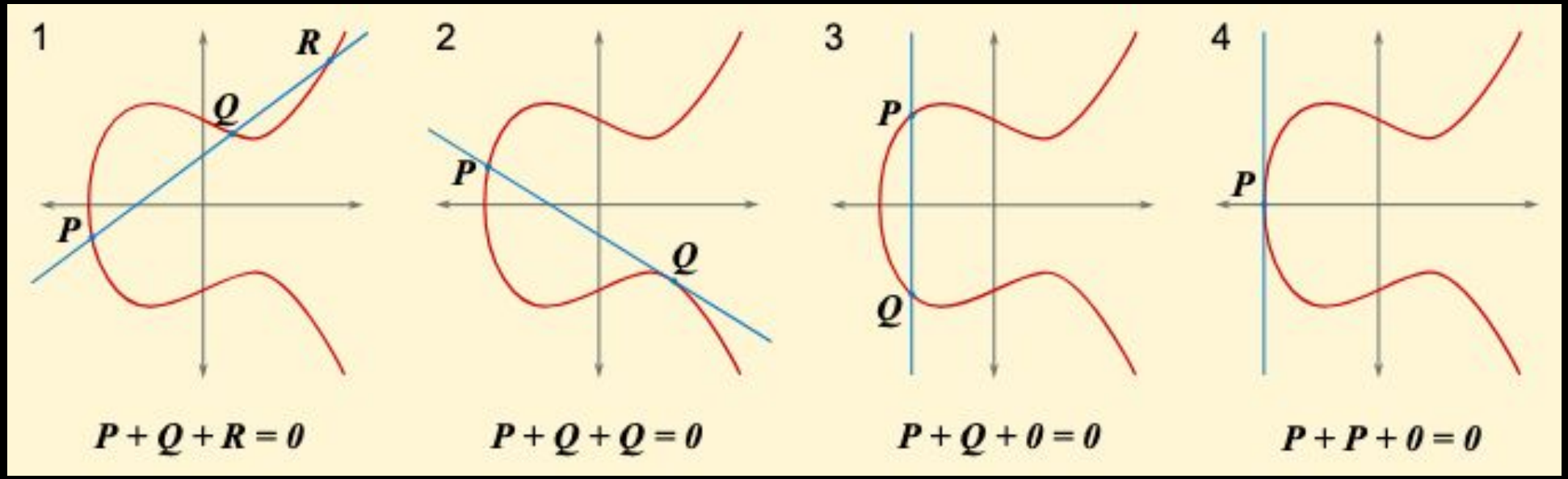
```
ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAIHy60AXoP5Qbn1oxSTB/Zy2Tl6NZDHjcv
36J70WsE4KY
```

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQDDD8NiqekaqR23MGmy/zEXMlc4b
CTMmXrlUdrI2mJk5mNqrRHKzJjrJTG2uqWwJldWtATFo7cb3sVUHrDZ3P
6wMK66pFN7PKHnzDUzqDuX/OBqzqCPgRgo/X8gSHgwLEzhYlAa+coZGP9
X1nsRjBsCSRZ9gThf478vtufVKVCaZIWejdXp5s4Wd3bUbfK7xL3frctl
i/hXla2e62jQk6hwmVNtf09NlEiXXFSTtt5KDJWKQNRyA+WsYXvQqzNuz
PjTsZY1JlTJLfzQ6QtLTT0dIoPVVPrS+/UvKoG2TOmVEq8uNwfpdPu6Ah
jlOlg1iJ0U3h/iQdaiAinzWdD9Rc0LGSJdq5ki6KPQ7iBjHq/X1IjIDdn
e9PT/HNUDc34m+hnjqeopkhKdfz/G+MutlZkTntqDdZDuILTvrUu2BMI5
XXbTFQeCr4VO6b9auGuZdpRlaUfJIDX35Z3p6ry6RWfv/vWv620pJP9Le
d/9F6/rrw47RFT8qnDKbkcgXqbU0c0=
```



Elliptic Curve Diffie Hellman

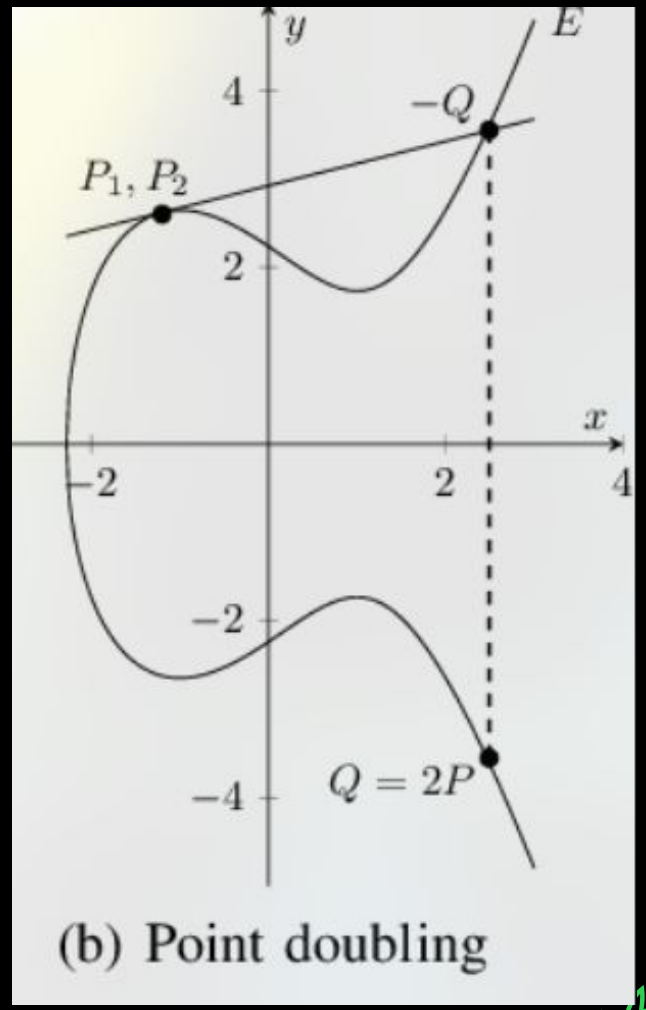
An *Elliptic Curve* is an algebraic curve where you can “add” points. That is, if you have points P and Q, you can get another point R such that addition satisfies all the usual laws.



Elliptic Curve Diffie Hellman

Repeated addition = multiplication

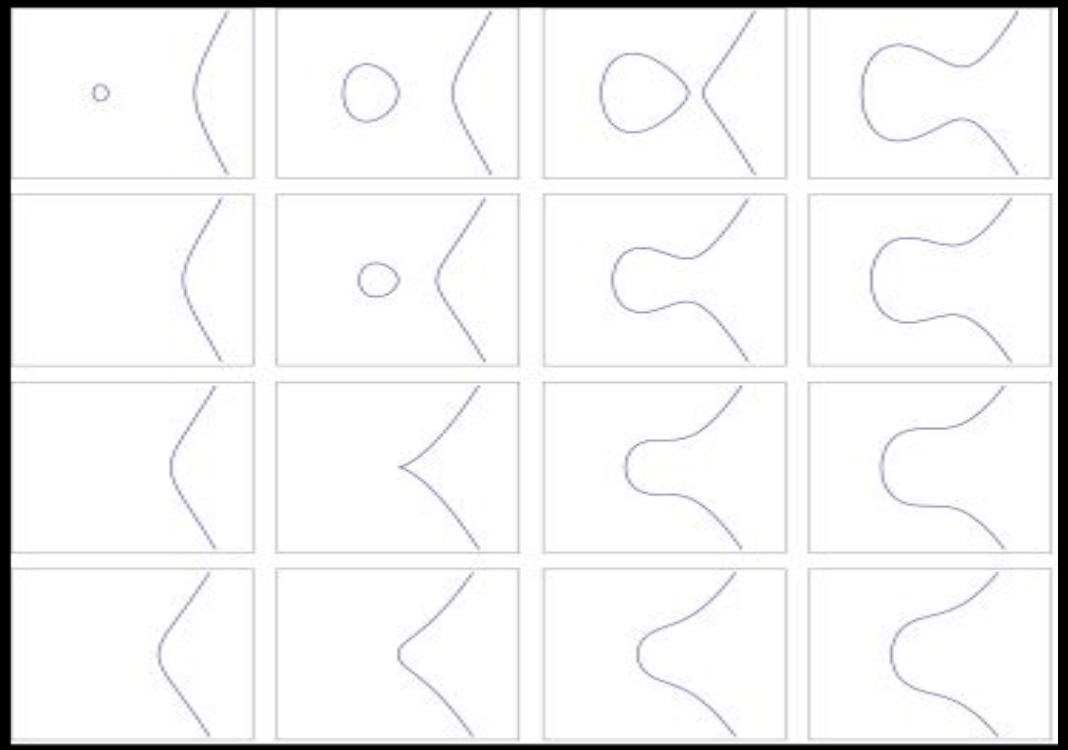
If we then repeatedly add a point to itself, we can calculate any integer scalar-multiplied point $Q = [x]P$ on the curve!



Elliptic Curve Diffie Hellman

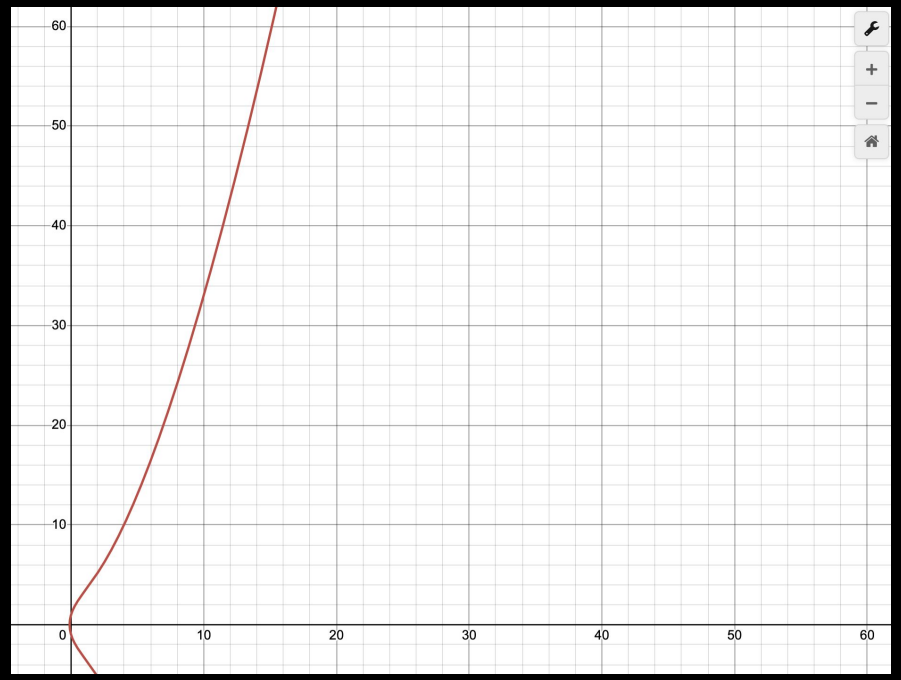
It turns out every such curve can be written in the form

$$Y^2 = X^3 + aX + b$$

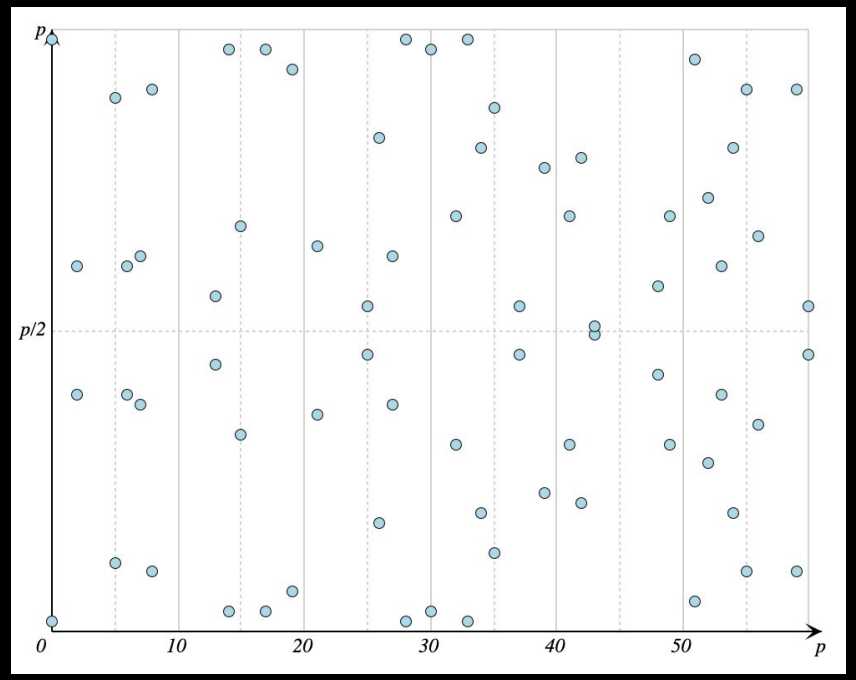


Elliptic Curve Diffie Hellman

Instead of considering the set of solutions to the curve $(x, y) \in \mathbb{R}^2$, only consider the set of solutions mod p O



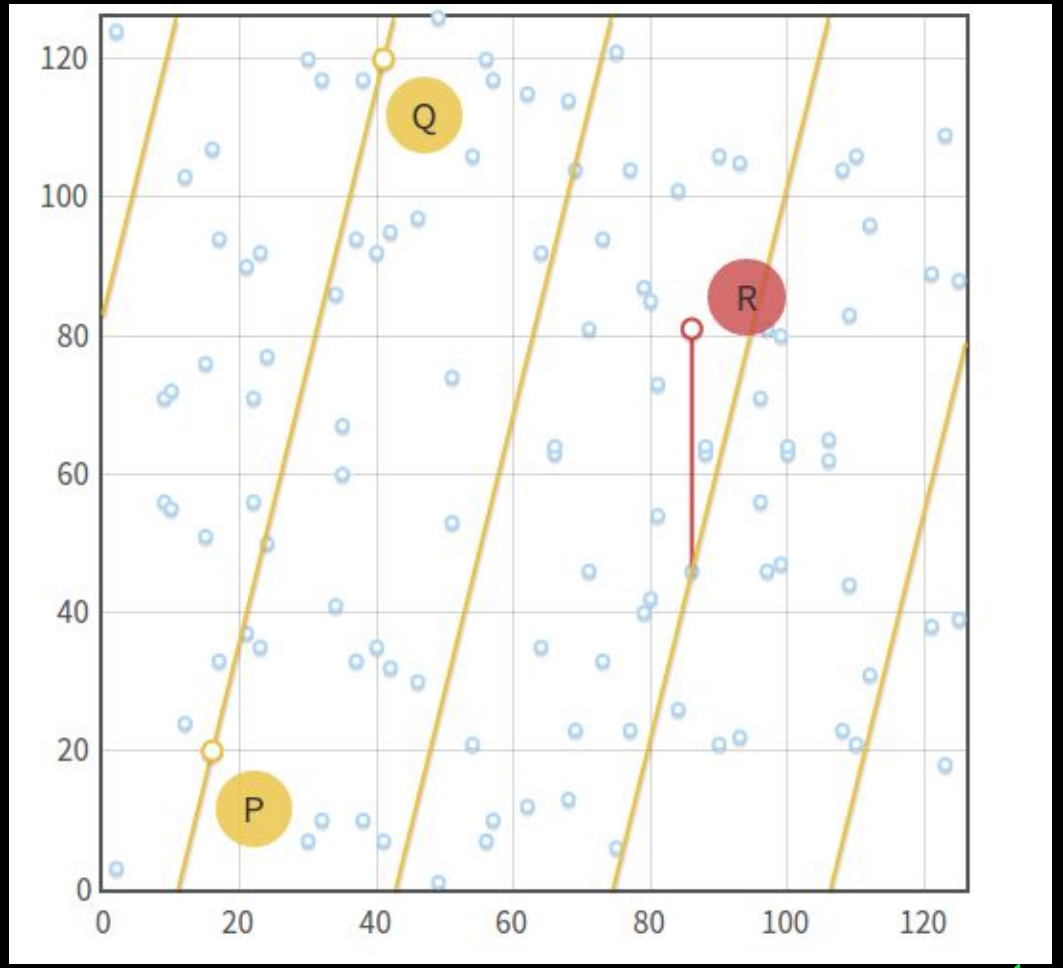
$$Y^2 = X^3 + 9X + 1$$



Same curve in $\mathbb{F}_{61}$

Elliptic Curve Diffie Hellman

- Alice and Bob share G and a fixed curve (Curve25519)
- Alice and Bob generate random a, b mod p as private keys
- They compute and share aG, bG as public keys
- Then the shared secret is a(bG) = b(aG) = (ab)G



Why a curve?

If we want to be able to represent everything on a computer, we want to represent things as numbers mod p.

The most obvious way to do this is to require everything to be a structure described by polynomials where we can multiply things.

It turns out the only 1-dimensional ways to do this are either normal DH or ECDH!

You can go to higher dimensions but there are new attacks and few advantages.

See this blog post for more details!



ECDH Attacks

- There are many interesting attacks if you use a weak curve
  - https://github.com/elikaski/ECC_Attacks
- However, if you use Curve25519 then there are essentially no possible misconfigurations
- Until last week there wasn’t even a subexponential attack!



Post Quantum Cryptography

In 1994 Peter Shor invented a quantum algorithm which gives an efficient solution to RSA and ECDH.

Hence, we need new cryptographic algorithms to solve this!



Q != PQ

Quantum cryptography is NOT the same as Post Quantum Cryptography.

Post Quantum Cryptography is Cryptography that is still secure from quantum algorithms. It does not need knowledge or usage of quantum computers.

Quantum Cryptography is using Quantum Computers to do cryptography. The advantage is that its physically guaranteed to work, but it is currently impractical.



Post Quantum Cryptography

Type	Examples	Pros	Cons
Lattices	Kyber	Decent key size, fast, least sus	New attacks
Codes	McEliece	Decent speed, very old	Huge key size, sus due to new attacks
Isogenies	SIKE, CSIDH	Excellent key sizes	Slow although improving, sus



LWE

LWE is the problem of finding some vector s given approximate linear equations on s.

Some linear algebra turns this into an encryption algorithm.

$$
\begin{aligned}
14s_1 + 15s_2 + 5s_3 + 2s_4 &\approx 8 \pmod{17} \\
13s_1 + 14s_2 + 14s_3 + 6s_4 &\approx 16 \pmod{17} \\
6s_1 + 10s_2 + 13s_3 + 1s_4 &\approx 3 \pmod{17} \\
10s_1 + 4s_2 + 12s_3 + 16s_4 &\approx 12 \pmod{17} \\
9s_1 + 5s_2 + 9s_3 + 6s_4 &\approx 9 \pmod{17} \\
3s_1 + 6s_2 + 4s_3 + 5s_4 &\approx 16 \pmod{17} \\
&\vdots \\
6s_1 + 7s_2 + 16s_3 + 2s_4 &\approx 3 \pmod{17}
\end{aligned}
$$

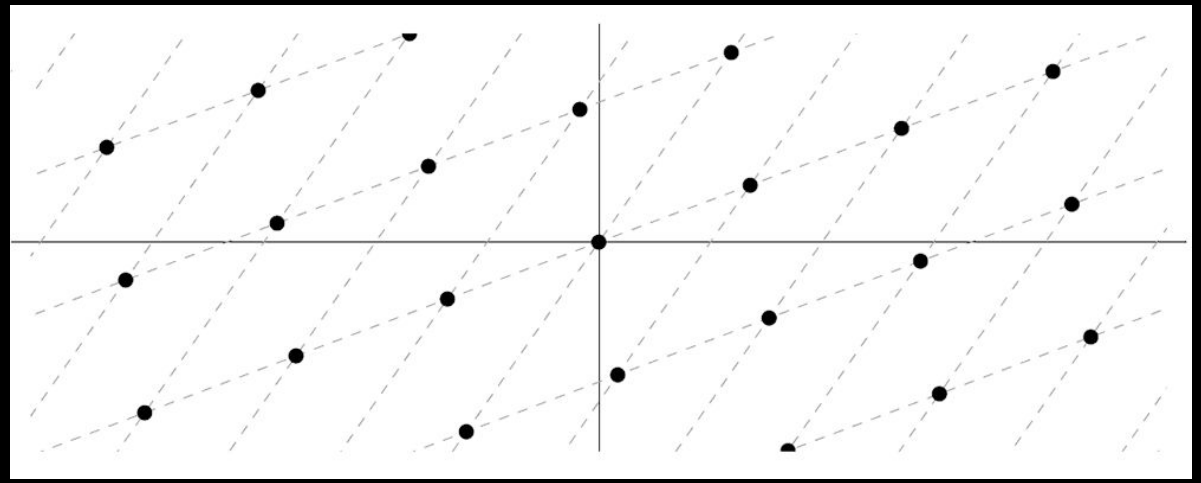


Why lattices?

Say the LHS of the previous equations are A*s, the RHS is t, and the error is e

Then t = A*s + e mod p, equivalently t = A*s + p*z + e

As z varies this forms lattice, and since e is small, finding A*s + p*z is the same as finding the closest vector to t, and finding s is easy

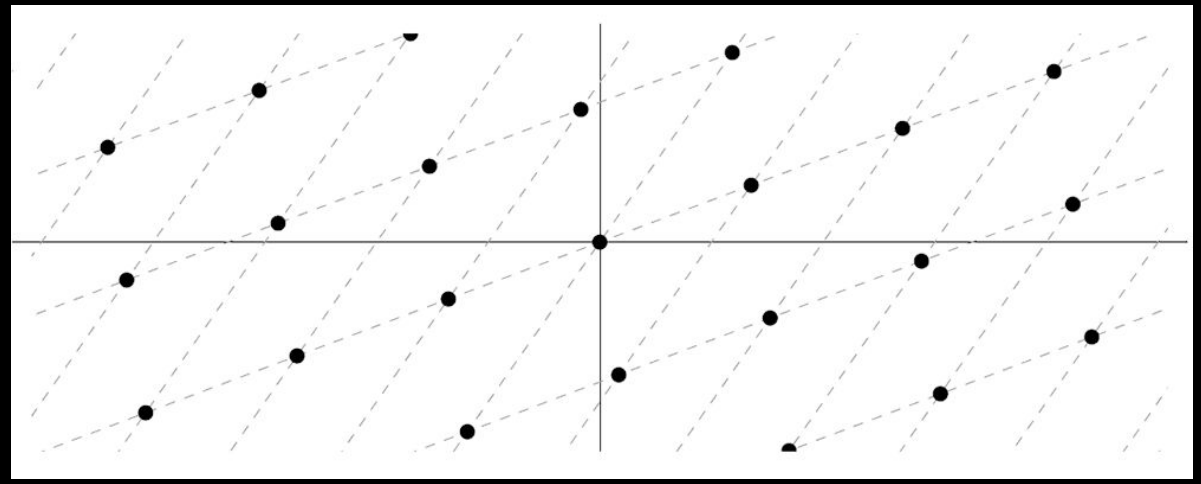




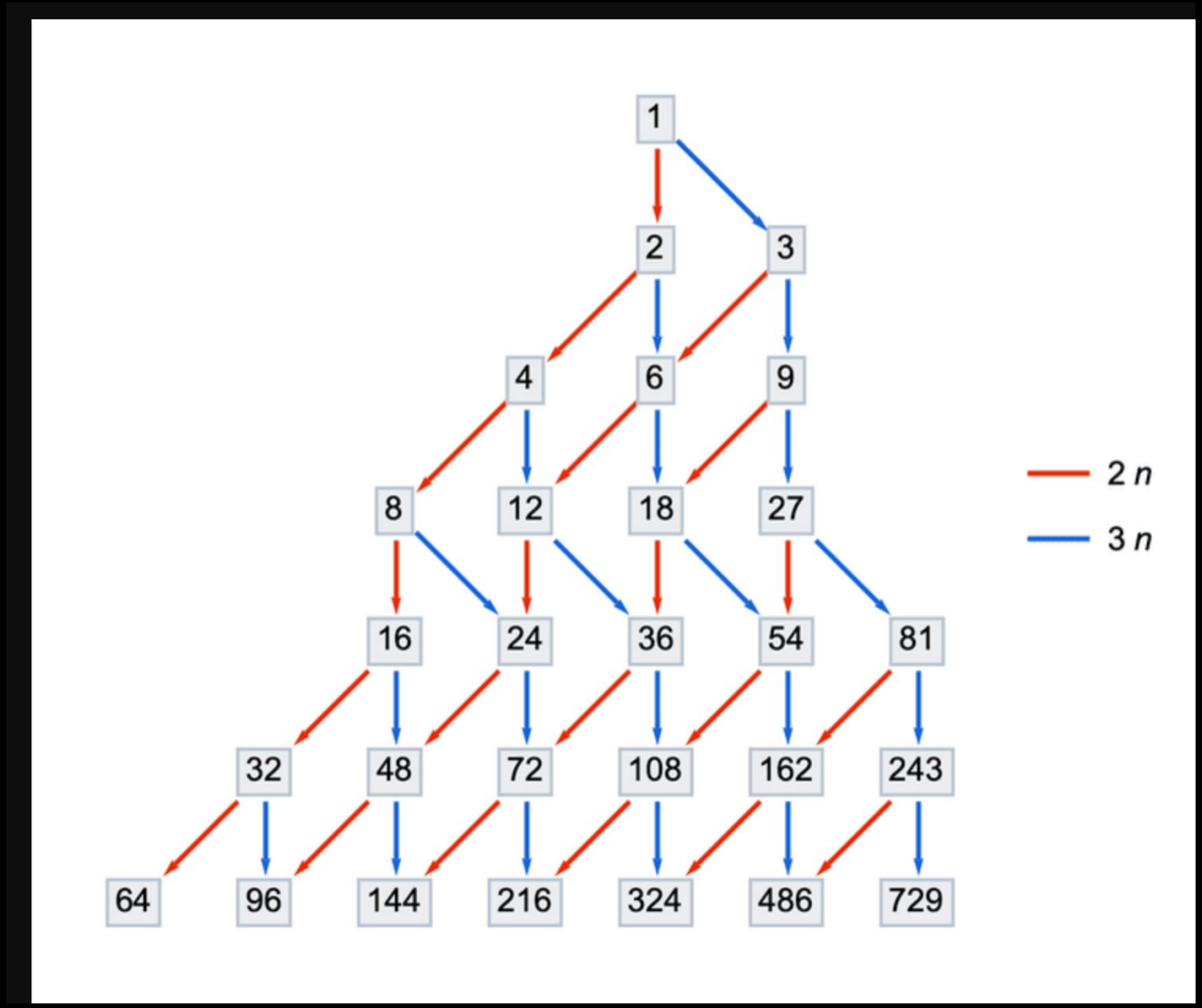
Why lattices?

The problem of finding the closest vector in a lattice to a point is known as the Closest Vector Problem (CVP).

This is NP-Complete, although we have good algorithms for low dimensions



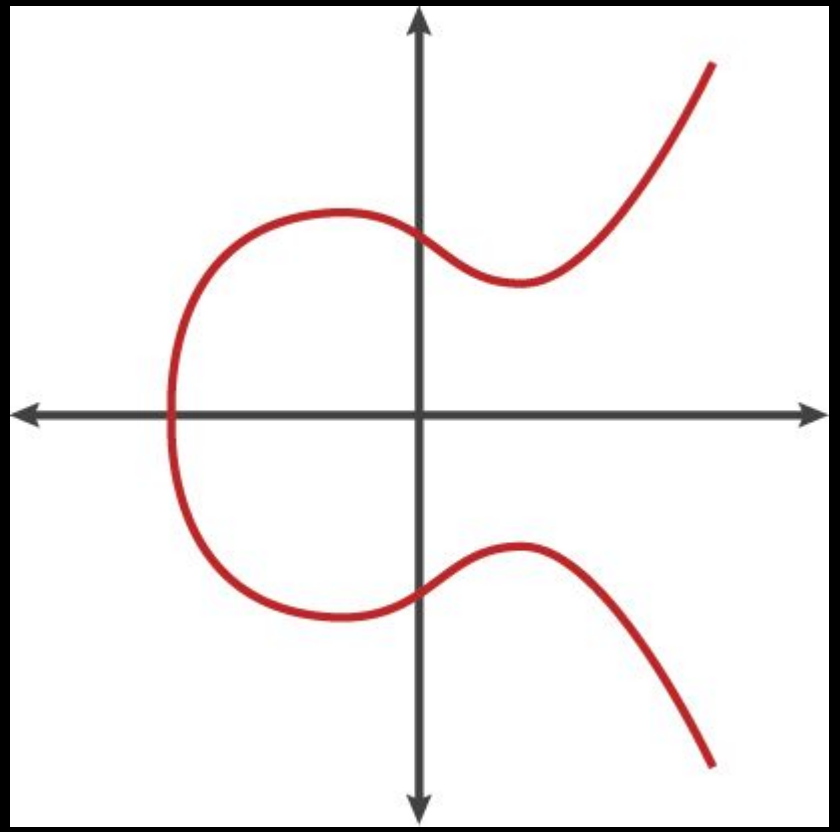
Isogenies


1
2
3
4
6
9
8
12
18
27
16
24
36
54
81
32
48
72
108
162
243
64
96
144
216
324
486
729
2 n
3 n

Isogenies

We want it to be hard to figure out the number associated to a vertex, but easy to traverse an edge.

We do this by representing each vertex by an Elliptic Curve.

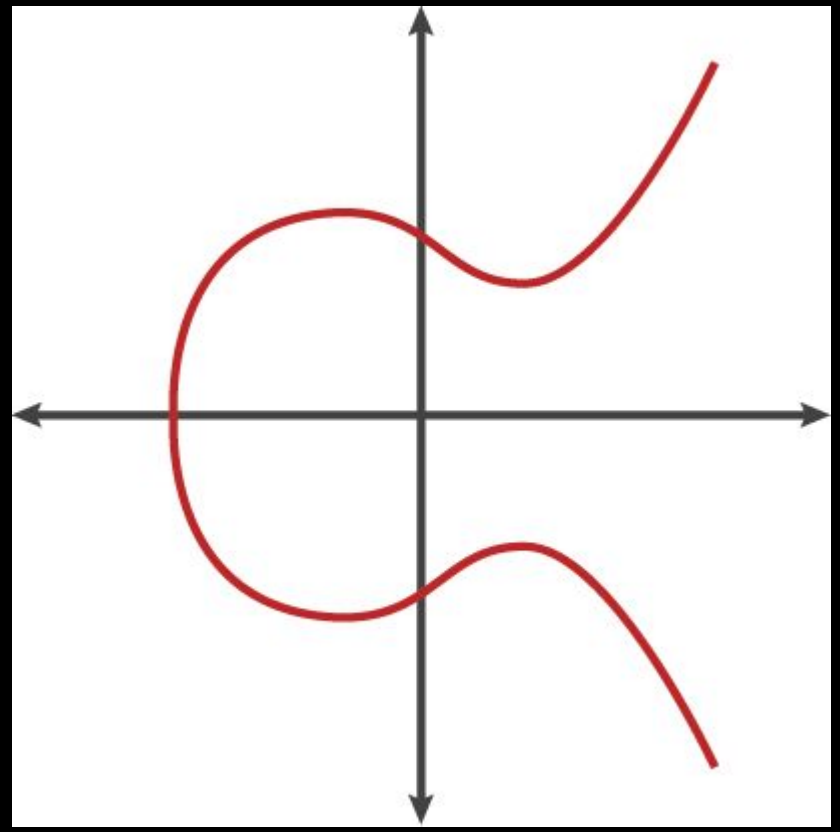


Isogenies

An edge of the graph is a function between the 2 curves known as an isogeny.

For example they should be continuous and f(P+Q) = f(P)+f(Q)

We have formulas to calculate the isogeny corresponding to any multiplication edge



Challenges

- Cryptohack!



Learn with fantastic lessons and challenges, and earn points on PwnyCTF while you're at it!

ctf.sigpwny.com/challenges#Meetings/CryptoHack

Next Meetings

`2025-10-24` • **This Friday**

- osu!gaming ctf
- Meet us in Siebel CS 2406 (may change, see Discord) for osu!gaming ctf!

`2025-10-26` • **This Sunday**

- Python Jails
- Learn about escaping python sandboxes to get arbitrary code execution on the main machine!



ctf.sigpwny.com

sigpwny{4}

Meeting content can be found at sigpwny.com/meetings.



SIGPwny