Diffie-Hellman: issues

- Diffie-Hellman is not secure against a MITM adversary
- Diffie-Hellman does not provide *authentication*
  - You exchanged keys with someone, but Diffie-Hellman makes no guarantees about who you exchanged keys with; it could be David!
- DHE is an *active protocol*: Alice and Bob need to be online at the same time to exchange keys
  - What if Bob wants to encrypt something and send it to Alice for her to read later?

TLS 1.3

Diffie-Hellman Key Exchange: Summary

P2P

- Algorithm:
  - Alice chooses X_A and sends $\alpha^{X_A} \; mod \; q$ to Bob
  - Bob chooses X_B and sends $\alpha^{X_B} \; mod \; q$ to Alice
  - Their shared secret is $(\alpha^{X_A})^{X_B} = (\alpha^{X_B})^{X_A} = \alpha^{X_A X_B} \; mod \; q$
- Diffie-Hellman provides forwards secrecy: Nothing is saved or can be recorded that can ever recover the key
- Diffie-Hellman can be performed over other mathematical groups, such as elliptic-curve Diffie-Hellman (ECDH)
- Issues
  - **Not** secure against MITM
  - Does not provide authenticity
  - Both parties must be online

DHKE in Python Cryptography Library

- https://cryptography.io/en/latest/hazmat/primitives/asymmetric/

Take home exercises

- SW, "Network Security Essentials", 6th Edition, 2017
  - Problems – 3.21

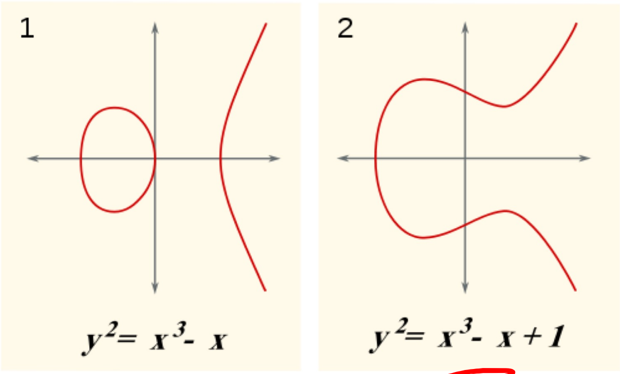
  Consider a Diffie-Hellman scheme with a common prime q = 11 and a primitive root α = 2.

  a. if user A has public key Y_A = 9, what is A's private key X_A?

  b. If user B has public key Y_B = 3, what is the shared secret key K?

Elliptic Curve Cryptography (ECC)

- 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



CPSC 467: Cryptography and Computer Security , Michael J. Fischer, 2017,
https://zoo.cs.yale.edu/classes/cs467/2017f/lectures/ln13.pdf

The Elliptic Curve Equation

An elliptic curve over real numbers:

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

where:

- $4a^3 + 27b^2 \neq 0$ (to avoid singularities)

Example curve:

$$y^2 = x^3 - 4x + 1$$

⚙️ Common ECC Curves

Curve	Field	Bit Strength	Usage
secp256k1	$\mathbb{F}_p$	128-bit	Bitcoin
P-256	$\mathbb{F}_p$	128-bit	TLS, NIST
Curve25519	$\mathbb{F}_p$	128-bit	Signal, SSH

Why ECC?

- In case of ECC, we are able to use smaller primes, or smaller finite fields, and achieve a level of security comparable to that of RSA
- ECC has 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.

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

ECC Key Generation

private key   generator $PK = k \cdot G$

- Let k be an integer and G a point on E. k×G is defined as adding G to itself k times.
- Once we calculate Q = k × G, it is extremely difficult to recover k from Q. The only way to recover k from Q is to try every possible repeated addition of G.

G

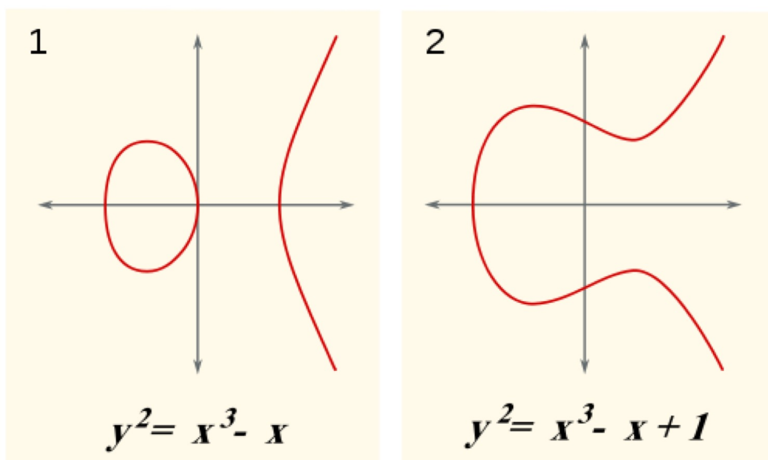
$e \cdot d \equiv 1 \mod \phi(n)$

- Q: Does it sound familiar?

Elliptic Curve Discrete Logarithm Problem (ECDLP)

PK.

- Let G be a point on E. Compute $Q = k \times G$. Then, ECDLP: given G and Q compute k. ← private key
- This allows us to translate crypto schemes based on DLP to EC-based schemes.
- Q is a public key. k is a private key. G is a generator point on E.



K , G

$Q = K \cdot G$

🚀 ECC in Practice

- **TLS 1.3:** Uses ECDHE for forward secrecy.
- **Cryptocurrencies:** Bitcoin, Ethereum (secp256k1).
- **Messaging apps:** Signal, WhatsApp (Curve25519).
- **Government/NIST:** P-256, P-384, P-521.

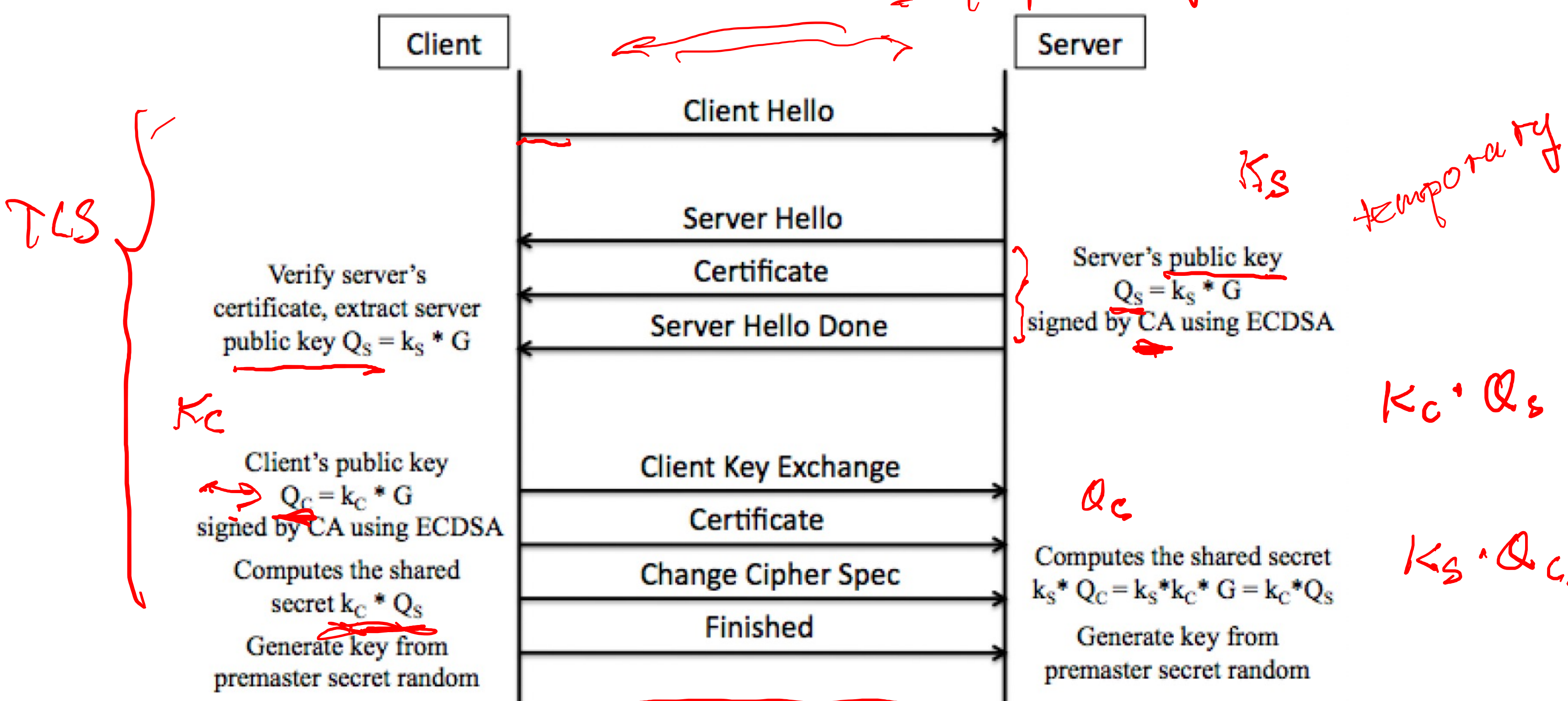
1.3

TLS 1.2

TLS Key Agreement with ECDH

3 way hands

TCP



Encrypted Data Https

RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3, https://datatracker.ietf.org/doc/html/rfc8446

Summary

- ECC achieves **strong security with small keys**.
- Based on the **hardness of ECDLP**.
- Powers many modern systems (TLS, blockchain, mobile apps).