Lecture 10: Public Key Cryptosystems based on Discrete Logarithms

TTM4135

Relates to Stallings Chapter 10

Spring Semester, 2025

Motivation

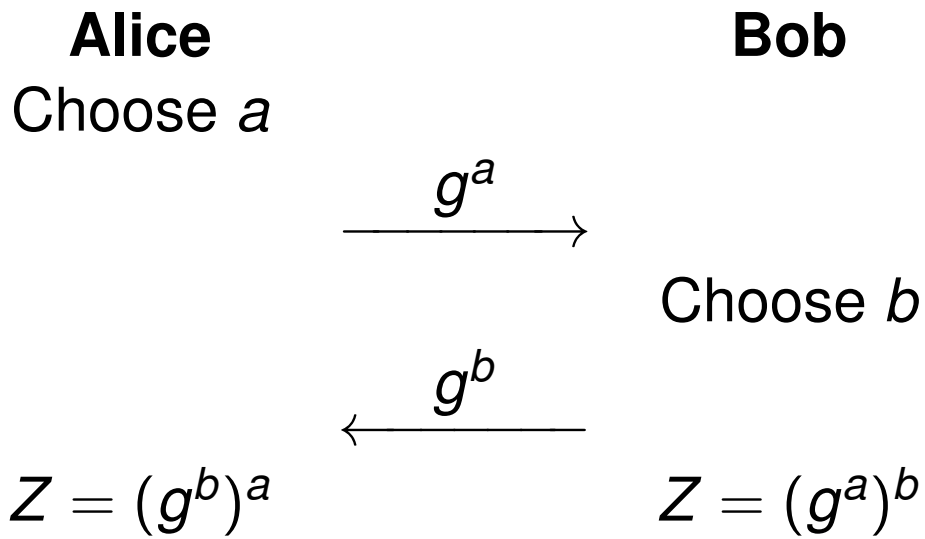
- Discrete log ciphers are currently the main alternative public key systems to RSA
- Discrete log ciphers are widely deployed and standardised
- When implemented on elliptic curves, discrete log ciphers are often more efficient than RSA

Outline

Diffie–Hellman key exchange

- Two users, Alice and Bob, want to share a secret using only public communications
- Public knowledge: generator g of a multiplicative group G of order t
- Alice and Bob each select random values a and b respectively where $0 < a, b < t$
- Alice sends g^a to Bob (*over an insecure channel*)
- Bob sends g^b to Alice (*over an insecure channel*)
- Alice and Bob both compute secret key $Z = g^{ab}$
- Originally group G used was $\mathbb{Z}_p^*$ for large p
- Today common to use elliptic curve group for G (see later in this lecture)

Diffie-Hellman Protocol



The shared secret value Z can be used to compute a key for, say, AES. This is done using a *key derivation function* based on a public hash function.

Example in $\mathbb{Z}_p^*$

Public knowledge is $p = 181, g = 2$

- Alice selects $a = 50$
- Bob selects $b = 33$
- Alice sends $g^{50} \bmod 181 = 116$ to Bob
- Bob sends $g^{33} \bmod 181 = 30$ to Alice
- Alice computes $Z = 30^{50} \bmod 181$
- Bob computes $Z = 116^{33} \bmod 181$
- Common secret is $Z = 49$

Security of Diffie–Hellman

- An attacker who can find discrete logarithms in G can break the protocol:
  1. intercept the value g^a and take the discrete log to obtain a
  2. compute g^{ab} in the same way as B
- It is unknown whether a better way exists to break the protocol than by taking discrete logs
- The problem of finding $Z = g^{ab}$ from knowledge of g^a and g^b is known as the *Diffie–Hellman problem*

Authenticated Diffie–Hellman

- In the basic Diffie–Hellman protocol the messages between Alice and Bob are not authenticated
- Neither Alice nor Bob knows who the secret Z is shared with unless the messages are authenticated
- This allows a *man-in-the-middle* attack, where the adversary sets up two keys, one with Alice and one with Bob, and relays messages between the two
- Authentication can be added in different ways, for example by adding *digital signatures* (see next lecture for how to construct digital signatures)

Static and ephemeral Diffie–Hellman

- The Diffie–Hellman protocol described above uses *ephemeral keys*: keys which are used once and then discarded
- In *static* Diffie–Hellman, Alice chooses a long-term private key x_A with corresponding public key $y_A = g^{x_A}$
- Similarly, Bob chooses a long-term private key x_B with corresponding public key $y_B = g^{x_B}$
- Now Alice and Bob can find a shared secret $S = g^{x_A x_B}$ just by looking up (or knowing beforehand) each others' public keys
- The secret S is static: it stays the same long-term, until Alice and Bob change their public keys

Elgamal cryptosystem

- Proposed by Taher Elgamal in 1985
- Turns the Diffie–Hellman protocol into a cryptosystem
- Here we look at original version where group G is $\mathbb{Z}_p^*$
- Alice combines her ephemeral private key with Bob's long-term public key

Elgamal key generation

- Select a prime p and a generator g of $\mathbb{Z}_p^*$
- Select a long term private key x where $1 < x < p - 1$
- Compute $y = g^x \bmod p$
- The public key is (p, g, y)
- Often p and g are shared between all users in some system

Encryption and decryption

Encryption The public key is $K_E = (p, g, y)$

1. For any value M where $0 < M < p$
2. Choose k at random and compute $g^k \bmod p$
3. $C = E(M, K_E) = (g^k \bmod p, My^k \bmod p)$

Decryption The private key is $K_D = x$ with $y = g^x \bmod p$

1. Let $C = (C_1, C_2)$
2. Compute $C_1^x \bmod p$
3. $D(C, K_D) = C_2 \cdot (C_1^x)^{-1} \bmod p = M$

Why it works

- The sender knows the ephemeral private key k and the long-term public key y
- The receiver knows the static private key x and the ephemeral public key $g^k \bmod p$
- Both sender and recipient can compute the Diffie–Hellman value for the two public keys: $C_1 = g^k \bmod p$ and $y = g^x \bmod p$
- This Diffie–Hellman value, $y^k \bmod p = C_1^x \bmod p$, is used as a mask for the message M

Powers of integers modulo 19

Table 2.7 Powers of Integers, Modulo 19

a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}	a^{18}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
3	9	8	5	15	7	2	6	18	16	10	11	14	4	12	17	13	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1
6	17	7	4	5	11	9	16	1	6	17	7	4	5	11	9	16	1
7	11	1	7	11	1	7	11	1	7	11	1	7	11	1	7	11	1
8	7	18	11	12	1	8	7	18	11	12	1	8	7	18	11	12	1
9	5	7	6	16	11	4	17	1	9	5	7	6	16	11	4	17	1
10	5	12	6	3	11	15	17	18	9	14	7	13	16	8	4	2	1
11	7	1	11	7	1	11	7	1	11	7	1	11	7	1	11	7	1
12	11	18	7	8	1	12	11	18	7	8	1	12	11	18	7	8	1
13	17	12	4	14	11	10	16	18	6	2	7	15	5	8	9	3	1
14	6	8	17	10	7	3	4	18	5	13	11	2	9	12	16	15	1
15	16	12	9	2	11	13	5	18	4	3	7	10	17	8	6	14	1
16	9	11	5	4	7	17	6	1	16	9	11	5	4	7	17	6	1
17	4	11	16	6	7	5	9	1	17	4	11	16	6	7	5	9	1
18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1

Stallings book, Table 2.7

Example

Key generation Choose prime $p = 181$ and generator $g = 2$

- Private key of A is $x = 50$
- Public key is $p = 181, g = 2, y = 116$

Encryption Sender wants to send $M = 97$

- Sender chooses random $k = 31$
- Ciphertext is $(98, 173) = (C_1, C_2)$

Decryption A receives (C_1, C_2) and recovers M by:

- $C_1^x = 98^{50} \bmod p = 138$
- $M = C_2 \times (C_1^x)^{-1} \bmod p$
  $= 173 \times 138^{-1} \bmod 181$
  $= 97$

Security of Elgamal

- If an attacker can solve the discrete log problem, the system can be broken by determining the private key x from $g^x \bmod p$
- It is quite possible for many users to share the same p and g values.
- There is no need for any padding as in RSA - notice that each ciphertext is already randomised
  - For the encryption scheme itself – may be required for some other properties
- The Elgamal cryptosystem has a proof of security in a suitable model subject to the difficulty of the so-called *decision Diffie–Hellman problem*.

Discrete Log Problem over $\mathbb{Z}_p$

- Let p be a large prime.
- We will let g denote a generator of the multiplicative group of $\mathbb{Z}_p$
- i.e. for a any nonzero element in $\mathbb{Z}_p$ we can find a unique i between 1 and $p-1$ such that $a = g^i \pmod{p}$.

We say that $\log_g(a) = i \pmod{p}$.

Discrete Log problem

The discrete log problem is:

$$\text{given } a, \text{ find } \log_g(a) \pmod{p}.$$

For sufficiently large p this is an intractable problem.

Using Discrete Log

We shall show that several public key ciphers can be based on the discrete log problem.

It should be noted that the implementation of these ciphers depends on the property that $a^{p-1} \equiv 1$ for all nonzero elements in $\mathbb{Z}_p$.

- If it is possible to compute discrete logs in G, then Decision Diffie-Hellman does not hold.

Discrete Log Problem and Factoring

Solving the discrete log problem over $\mathbb{Z}_p$ is comparable to the difficulty of factoring n, where n is the product of two primes, i.e. if the number of bits in n is the same as the number of bits in p. Hence, the discrete log ciphers modulo p offer the same level of security as the RSA algorithm.

What are elliptic curves?

- Elliptic curves are algebraic structures formed from cubic equations
- An example is the set of all (x, y) pairs which satisfy the equation:

$$y^2 = x^3 + ax + b \bmod p$$

- This example is a curve over the field $\mathbb{Z}_p$ but elliptic curves can be defined over any field
- Once an identity element is added, a binary operation (like addition or multiplication) can be defined on these points
- With this operation the elliptic curve points form an *elliptic curve group*

An example

- Recall $y^2 = x^3 + ax + b \bmod p$. Call this curve $E_{23}(1, 1)$.
- Let $a = b = 1$, and $p = 23$, so we look at the curve $y^2 = x^3 + x + 1$.
- We can show that $x = 9$ and $y = 7$ satisfies the equation.
  - $7^2 = 49 \equiv 3 \bmod 23$.
  - $9^3 + 9 + 1 \equiv 3 \bmod 23$.
  - So then indeed, we have that $7^2 \equiv 9^3 + 9 + 1 \bmod 23$.
- Therefore, the point $P = (9, 7)$ is on the curve.

Other points on $E_{23}(1, 1)$

Table 10.1 Points (other than O) on the Elliptic Curve $E_{23}(1, 1)$

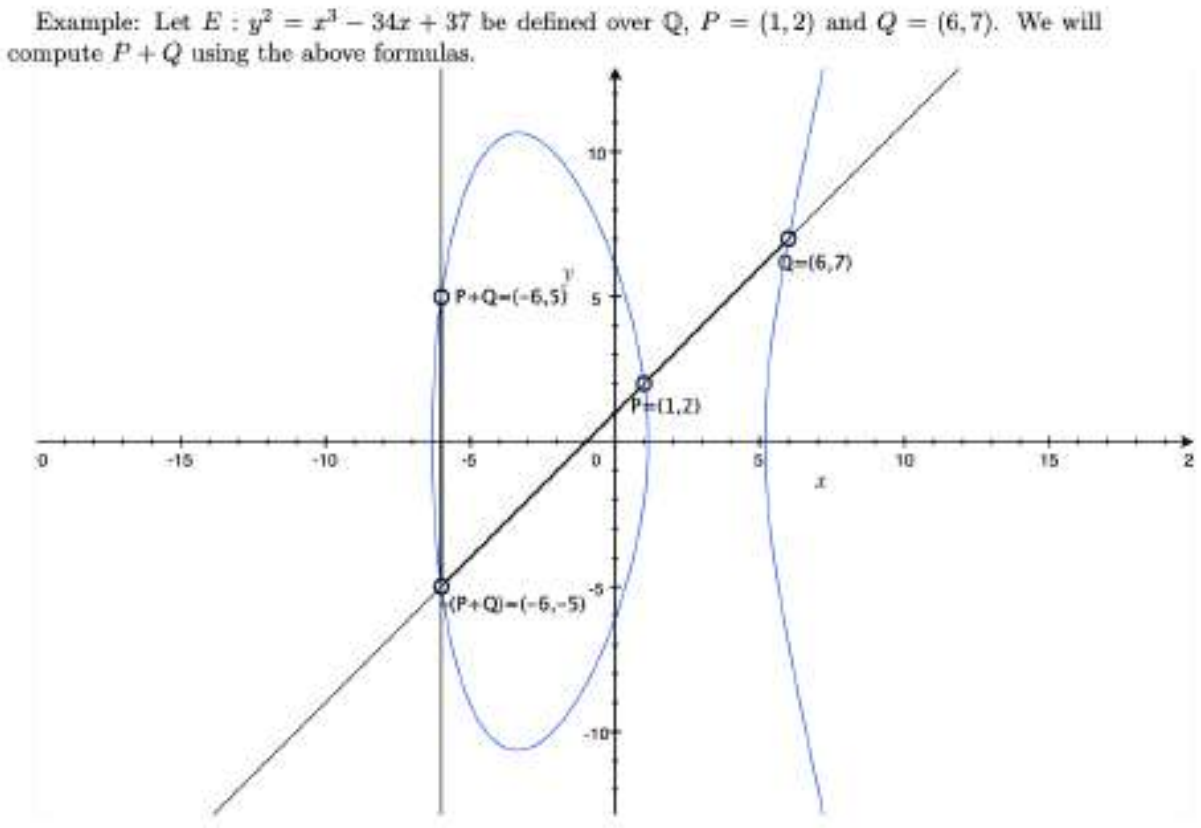
(0, 1)	(6, 4)	(12, 19)
(0, 22)	(6, 19)	(13, 7)
(1, 7)	(7, 11)	(13, 16)
(1, 16)	(7, 12)	(17, 3)
(3, 10)	(9, 7)	(17, 20)
(3, 13)	(9, 16)	(18, 3)
(4, 0)	(11, 3)	(18, 20)
(5, 4)	(11, 20)	(19, 5)
(5, 19)	(12, 4)	(19, 18)

Stallings book, Table 10.1

Elliptic curve computations

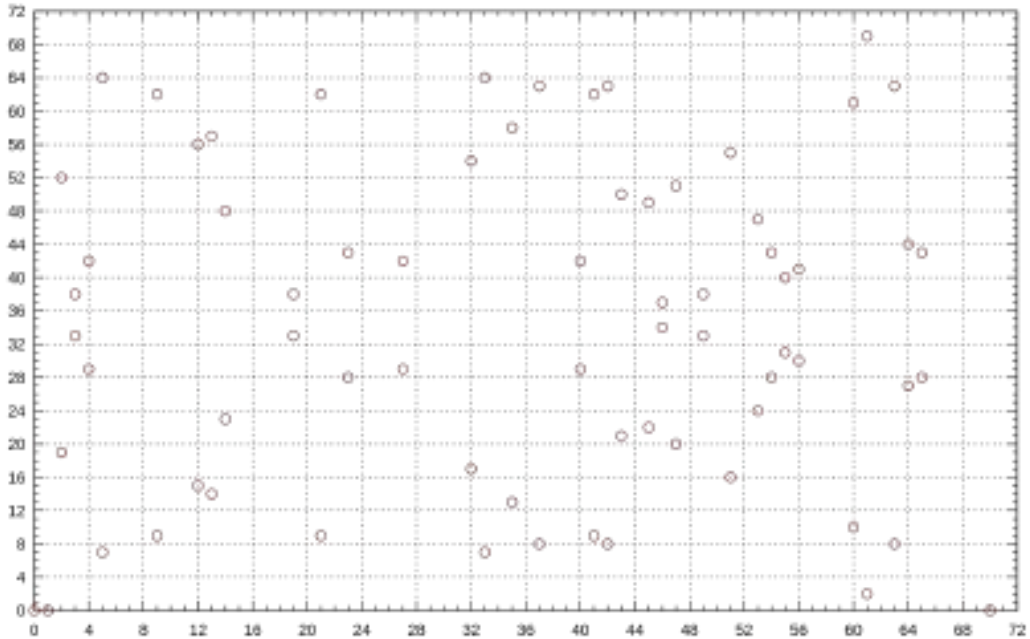
- The elliptic curve group operation could be denoted by any symbol, but by convention is it usually called elliptic curve *addition*
- We write $P + Q = R$ to show the group operation on curve points P and Q with result R
- The *elliptic curve discrete log problem* is to find the value of m, given a point P and a generator G so that $P = mG = G + G + \ldots + G$ (m times)
- Efficient computation of elliptic curve multiplication can use the *double-and-add algorithm*, by replacing multiplication in the square and multiply algorithm with addition

Elliptic Curve – visual representation



https://www.umsl.edu/~siegelj/information_theory/projects/elliptic_curves_group_law.pdf

Elliptic Curve – visual representation



Representation of $y^2 = x^3 - x$ over the finite field $\mathbb{F}_{71}$

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

Elliptic curve representations

There are several different ways of representing elliptic curves which use different point formats and different ways to computing the group operation

Short Weierstrass form is the most common format as shown on Slide 21

Montgomery form allows a fixed time elliptic curve multiplication which is useful to avoid timing side channels

Edwards form allows for faster group operations

It is common to shift between representations to optimise performance

Choosing elliptic curves

- A new elliptic curve can be generated at any time, but standard applications usually use standard curves
- Various predefined sets of curves exist such as the set of NIST curves contained in the standard SP 800-186 (2023)
- Desirable that standard curves are generated in a way to ensure there are no hidden special properties but researchers have disputed this for some standard curves

Example - NIST curve P-256

$$
\begin{aligned}
p &= \texttt{1157920892103562487626974469494075735300861434152903141955336313088670978539 51} \\
n &= \texttt{1157920892103562487626974469494075735299969552241357603424222590610685120443 69} \\
s &= \texttt{3045ae6fc8422f64ed579528d38120eae12196d5} \\
c &= \texttt{3099d2bbbfcb2538542dcd5fb078b6ef5f3d6fe2c745de65} \\
b &= \texttt{4105836372515214212932612978004726840911444101599372555483525631403946740129 1} \\
G_x &= \texttt{4843956129390645175905258525279791420 2762949526041747995844080717082404635286} \\
G_y &= \texttt{3613425095674979579858512791958788195 6 611106672985015071877198253568414405109}
\end{aligned}
$$

- Curve of n points (256-bits) over $\mathbb{Z}_p$ with generator (G_x, G_y) and equation: $y^2 = x^3 - 3x + b \bmod p$
- Parameter s is the seed for the random generation and c is the output of a SHA-1 hash generated from s

Curve 25519

- A curve allowing very fast computations
- Proposed by Bernstein in 2005 and now a NIST recommended curve
- Equation of the curve is $y^2 = x^3 + 486662x^2 + x$ which is in Montgomery form
- The field for computations is the integers modulo p, where $p = 2^{255} - 19$ is prime

Discrete logarithms on elliptic curves

- The discrete logarithm on elliptic curve groups is to find the value of m, given a point P and a generator G so that $P = mG = G + G + \ldots + G$ (m times)
- This is the same as in $\mathbb{Z}_p^*$ but with addition as the group operation instead of multiplication
- The best known algorithms for solving the elliptic curve discrete log problem are *exponential* in the length of the parameters
  - Subexponential algorithms exist for factoring and finite field discrete log
- Consequently elliptic curve implementations use much smaller keys
- Compared with RSA the relative advantage of elliptic curve cryptography will *increase* at higher security levels

Comparing strength of elliptic curve cryptography

Symmetric key length	RSA modulus length or length of p in $\mathbb{Z}_p^*$	Elliptic curve group size
80	1024	160
128	3072	256
192	7680	384
256	15360	512

- For example, brute force search of 128-bit key for AES takes roughly same computational effort as for taking discrete logarithms in $\mathbb{Z}_p^*$ with p of 3072-bits or on an elliptic curve with elements of size 256 bits
- Source: adapted from NIST SP 800-57 Part 1, Recommendations for Key Management (revised 2020)

Elliptic curve cryptography

- Most cryptosystems based on discrete logarithms can be constructed with elliptic curves as well as in $\mathbb{Z}_p^*$
- In particular, Diffie–Hellman key exchange and Elgamal encryption can be run on elliptic curves
- Elliptic curve cryptography is today widely deployed
- The Canadian company Certicom, now part of Research in Motion, holds many patents on some of the mathematical ideas

Post-quantum cryptography

- Most public key cryptography is use today will be broken if quantum computers become available due to Shor's algorithm for factorisation, which can also be used to find discrete logarithms
- *Post-quantum cryptography* is concerned with building cryptographic primitives which will remain secure if this happens
- Symmetric key cryptography can still be used but with double length keys due to Grover's algorithm for searching
- NIST process to standardise post-quantum cryptography has recommended initial algorithms but is still continuing

Post-quantum Diffie–Hellman

- Post-quantum secure cryptosystems are based on different problems, particularly lattice problems, coding theory, and solving multi-variable polynomials
- Currently we do not have a post-quantum “drop-in” replacement for Diffie–Hellman
- A promising candidate was the use of isogenies on elliptic curves
  - But this was broken, see: https://ellipticnews.wordpress.com/2022/08/12/attacks-on-sidh-sike/