Department of Mathematics
Stockholm University
Section: Mathematics
Examiner: Olof Sisask

Exam
Mathematics of Cryptography
MM7018, 7.5 hp
2020-03-16 09:00–14:00

There are ten problems, each giving between 0 and 8 points. The points from the exam are added to your points from the homework assignments. Grades are then given by the following intervals:

A: 100–92p, B: 91–84p, C: 83–76p, D: 75–68p, E: 67–60p.

Remember to motivate your answers carefully. You may use Big-O notation unless explicit constants are asked for. No calculators or computers may be used.

Try to keep your answers concise; no question is asking for an essay.

1. (a) (2p) Describe the basic framework for a private key (symmetric) cryptosystem.
   (b) (2p) Describe the basic framework for a public key cryptosystem. Mention, in particular, what a one-way function with trapdoor information is, and its role in the system.
   (c) (1p) State some of the main advantages of public key cryptosystems relative to private key cryptosystems, and vice versa. (Short explanations suffice.)
   (d) (3p) Explain the essential properties that a practical cryptosystem must have when it comes to speed of computation. What does it mean for a system to be secure against a *known plaintext attack*? What does it mean to be secure against a *chosen plaintext attack*? (You may answer in the context of either symmetric or asymmetric cryptosystems.)

Solution (a) A private key cryptosystem consists of

- a set M of allowable plaintexts,
- a set C of ciphertexts,
- a set K of keys and,
- for each key $k \in K$, a pair of functions $e_k : M \to C$ and $d_k : C \to M$ such that $d_k(e_k(m)) = m$ for all $m \in M$.

The functions e_k are called encryption functions and the d_k are decryption functions.

(b) Similar, except that the set of keys consists of pairs $k = (k_{\text{priv}}, k_{\text{pub}})$ of private and public keys, and the encryption functions depend only on the public key, so $e_k = e_{k_{\text{pub}}}$ and the decryption function depends on the private key.

One should take each $e_{k_{\text{pub}}}$ to be a one-way function, that is an invertible function whose inverse is hard to compute without knowledge of the corresponding k_{priv}, and for which the inverse is easy to compute when in possession of k_{priv}. The private key k_{priv} is also known as trapdoor information.

(c) Advantages of public key cryptosystems: the main advantage is that there is no need to exchange a key beforehand or to exchange any information in secret. It is also easy to change key often.

The main advantages of private key systems are that they are usually faster, and that both communicating parties have symmetric encryption and decryption capabilities.

(d) It must be

- fast to encrypt (using the encryption key),
- fast to decrypt if you have the decryption key,
- slow to decrypt, by any means, without knowledge of the decryption key.
- Secure against known plaintext attack: If you know a pair (c, m) consisting of a ciphertext and its corresponding plaintext, then it should still be hard to decrypt other ciphertexts. Similarly if you know some fixed number of such pairs.

- Secure against chosen plaintext attack: If you are allowed to pick a plaintext m and get told its corresponding ciphertext c, then it should still be hard to decrypt any ciphertexts other than c. Similarly if you are allowed to pick a few plaintexts $m_1, m_2, \ldots$

2. (a) (1p) Show the steps used by the Euclidean algorithm to compute $\gcd(55, 34)$.
   (b) (2p) Describe the Euclidean algorithm for computing $\gcd(a, b)$, where $a \geq b \geq 1$ are integers.
   (c) (3p) Prove that the algorithm terminates in at most $2 + 2\log_2 b$ division steps.
   (d) (2p) Prove that the algorithm needs at least $\log_2 b$ division steps for infinitely many integers b. Hint: let $a_0 = 1$, $a_1 = 1$ and $a_{k+1} = a_k + a_{k-1}$, for $k \geq 1$, be the Fibonacci sequence. Analyse how the Euclidean algorithm performs when computing $\gcd(a, b)$ for $a = a_{n+1}$, $b = a_n$. (You do not need to use the most accurate relationship between n and b to prove the stated bound; a simple one will do.)

Solution (a) —

(b) One successively performs division with remainder until one hits a remainder of 0, in the following manner:

$$\begin{aligned} a &= q_1 b + r_2 \\ b &= q_2 r_2 + r_3 \\ r_2 &= q_3 r_3 + r_4 \\ r_3 &= q_4 r_4 + r_5 \\ &\vdots \\ r_{n-2} &= q_{n-1} r_{n-1} + r_n \\ r_{n-1} &= q_n r_n + 0, \end{aligned}$$

where $r_j \neq 0$ for all $j < n$. Then $\gcd(a, b) = r_n$.

(c) We claim that the remainders r_i decrease by a factor at least 2 every two steps. Let $r_0 = a$ and $r_1 = b$. Claim: for $0 \leq i \leq n-2$ we have

$$r_{i+2} < \tfrac{1}{2} r_i.$$

Proof: the remainders r_j are clearly decreasing, ie $r_{j+1} < r_j$, so if $r_{i+1} \leq \frac{1}{2} r_i$, then $r_{i+2} < r_{i+1} \leq \frac{1}{2} r_i$, and we are done. Otherwise, if $r_{i+1} > \frac{1}{2} r_i$, then the quotient in the division of r_i by r_{i+1} must be 1, and so

$$r_i = r_{i+1} + r_{i+2} \implies r_{i+2} = r_i - r_{i+1} < \tfrac{1}{2} r_i.$$

In either case, the claim is proved.
Thus, provided $2k + 1 \leq n$,

$$b = r_1 > 2 r_3 > 2^2 r_5 > \cdots > 2^k r_{2k+1} \geq 2^k.$$

Taking $k = \lfloor (n-1)/2 \rfloor$, we see that

$$k \leq \log_2 b \implies n \leq 2 + 2\log_2 b.$$

(d) Let $a_0 = 1$, $a_1 = 1$ and $a_{k+1} = a_k + a_{k-1}$, for $k \geq 1$, be the Fibonacci sequence. Let $a = a_{n+1}$ and $b = a_n$. The output of the algorithm above is then

$$\begin{aligned} a_{n+1} &= a_n + a_{n-1} \\ a_n &= a_{n-1} + a_{n-2} \\ a_{n-1} &= a_{n-2} + a_{n-3} \\ &\vdots \\ a_3 &= a_2 + a_1 \\ a_2 &= 2a_1 + 0. \end{aligned}$$

It thus takes n steps to complete. Since $a_n \leq 2^n$, which is clear since $a_{k+1} = a_k + a_{k-1} \leq 2a_k$ and $a_1 = 1$, we have that the algorithm takes

$$n \geq \log_2 a_n = \log_2 b \text{ steps to complete.}$$

Since there are infinitely many choices of n there are infinitely many different integers b.

3. (a) (2p) Compute $3^{129} \pmod{6480}$. Give your answer as an integer in $\{0, 1, 2, \ldots, 6479\}$.
   (b) (2p) Compute $5^{601} \pmod{151}$. Give your answer as an integer in $\{0, 1, 2, \ldots, 150\}$. If you use a theorem, make sure you justify why its hypotheses are satisfied.
   (c) (2p) Determine all integer solutions to the system of congruences

$$\begin{cases} 4x \equiv 2 \pmod{15} \\ x \equiv 3 \pmod{49}. \end{cases}$$

   (d) (2p) The element 2 is a primitive root in $\mathbb{F}_{53}^{\times}$. Determine all integer solutions to

$$4^x \equiv 11 \pmod{53}.$$

Solution (a) Fast powering: 243

(b) Using Fermat's Little Theorem: since 151 is prime (not divisible by $2, 3, 5, 7$ or 11, and $13^2 > 151$), we know that $a^k \equiv a \pmod{151}$ if $k \equiv 1 \pmod{150}$. Since $601 \equiv 1 \pmod{150}$, we have

$$5^{601} \equiv 5 \pmod{151}.$$

(c) The second congruence is equivalent to $x = 3 + 49k$ for $k \in \mathbb{Z}$. The first congruence is then equivalent to

$$\begin{aligned} & 4(3 + 49k) \equiv 2 \pmod{15} \\ \iff & 12 + k \equiv 2 \pmod{15} \\ \iff & k \equiv 5 \pmod{15}. \end{aligned}$$

Thus the congruences are equivalent to $x = 3 + 49(5 + 15m)$ for $m \in \mathbb{Z}$, or

$$x = 248 + 15 \cdot 49m \quad \text{for } m \in \mathbb{Z},$$

or

$$x \equiv 248 \pmod{15 \cdot 49}.$$

(d) Since 2 is a primitive root, $\operatorname{ord}(2) = 52$, and so $\operatorname{ord}(4) = \operatorname{ord}(2^2) = 52/2 = 26$. One solution to the congruence is $x = 3$, since $4^3 = 64 \equiv 11 \pmod{53}$, and so all solutions are described by

$$x \equiv 3 \pmod{26}.$$

4. Let G be a finite group, and let $g \in G$.
   (a) (1p) Describe what is meant by a Discrete Logarithm Problem (DLP) to base g in G.
   (b) (3p) Describe Shanks's Babystep–Giantstep Algorithm for solving a DLP in G. (You do not need to prove that the algorithm works.)
   (c) (1p) In how many steps does this algorithm guarantee to solve the DLP? How much storage does it need in general? Relate your answers to part (b).
   (d) (1p) What is the main advantage of Pollard's ρ method over Shanks's method for solving the DLP in $\mathbb{F}_p^{\times}$? Is there a disadvantage?

(e) (2p) If the group is $(\mathbb{Z}/N\mathbb{Z}, +)$ (that is, with addition as the group operation) and g is relatively prime to N, explain how the DLP to base g can be solved very quickly.

Solution (a) A DLP to base g in G is a problem of the form: given $h \in G$, find an integer x such that

$$g^x = h,$$

if it exists.

(b) Suppose g has order N and that we wish to solve $g^x = h$. Let $n = \lfloor \sqrt{N} \rfloor + 1$. Compute the two lists

$$\begin{aligned} &\text{List 1:} \quad 1, g, g^2, g^3, \ldots, g^n \\ &\text{List 2:} \quad h, hg^{-n}, hg^{-2n}, hg^{-3n}, \ldots, hg^{-n^2}, \end{aligned}$$

each of length $O(\sqrt{N})$. If the DLP has a solution, then by construction there is guaranteed to be an element that is in both lists. Find such a collision, say

$$g^j = hg^{-kn}, \quad \text{and note that} \quad g^{j+kn} = h,$$

and so $x = j + kn$ is a solution to the DLP.

(c) It needs $O(\sqrt{N} \log N)$ steps; $O(\sqrt{N})$ to compute the two lists above, and the log-factor comes from sorting and searching. It requires $O(\sqrt{N})$ storage, to form the two lists.

(d) For Pollard's ρ method, the storage requirement is $O(1)$. The main disadvantage is that its analysis relies on probabilisic reasoning with a function that is not known to be sufficiently 'mixing'. The expected running time is otherwise $O(\sqrt{N})$.

(e) If the group is $(\mathbb{Z}/N\mathbb{Z}, +)$, the DLP $g^x = h$ becomes, in additive notation,

$$gx \equiv h \pmod{N}.$$

If g is relatively prime to N, we can find its multiplicative inverse g^{-1} quickly by the Extended Euclidean algorithm, and then simply compute

$$x \equiv g^{-1}h \pmod{N}.$$

The running time is then $O(\log N)$.

5. (a) (2p) Describe a practical use for digital signature schemes.

(b) (4p) Describe all the steps in any one specific digital signature scheme from the course.

(c) (2p) Describe what hashing is, and its role in digital signature creation.

Solution (a) For example in verifying that software updates downloaded to one's computer are actually from the original software authors. They can be used in any situation in which one needs to know that a presented document is approved of by a particular party.

(b) —

(c) Often the signatures produced are of the same size (at least) as the document needing to be signed. For large documents, this is impractical. Instead, one often computes a hash of the document, meaning feeding the document to a specific hashing function that outputs a short integer h (say 256 bits) for which it is computationally infeasible to come up with any other documents that map to the same hash h. One then signs this hash h instead. Since it is hard to come up with any other documents that map to the same hash, it is hard for any attackers to claim that any other documents have been signed with the given signature.

6. (a) (2p) State the Pohlig–Hellman theorem on the number of steps needed to solve a DLP to a base g, when the order N of g is known to factor as $N = q_1^{e_1} q_2^{e_2} \cdots q_t^{e_t}$, where the q_i are distinct primes. (Make sure you state in which setting the theorem applies.)

   (b) (2p) Suppose you wish to pick a prime p and a primitive root in $\mathbb{F}_p^\times$ for use as a base for a DLP. What implications does the Pohlig–Hellman theorem have for your choices if you want the DLP to be secure?

   (c) (4p) Let G be a group, and q a prime. Suppose you have an algorithm that can solve the DLP in G to any base of order q in S_q steps. Let $g \in G$ be an element of order q^2. Prove that you can solve any DLP in G to base g in $O(S_q)$ steps.

Solution (a) —

(b) One should pick p so that $p-1$ has at least one large prime factor. If $p-1$ only has small prime factors, then the Pohlig–Hellman theorem allows one to solve DLPs quickly in $\mathbb{F}_p^\times$.

(c) —

7. (a) (3p) Suppose Alice wants to receive secure, encrypted messages from others using the RSA cryptosystem. Describe what she needs to do to set this up, including how she can decrypt encrypted messages. Explain why the decryption process works. (You may assume theorems from modular arithmetic.)

   (b) (2p) What is the mathematical problem underpinning the security of RSA, according to the best approaches known? Specify the running time for one of the two best methods known for solving this problem. Classify whether it is exponential, subexponential or polynomial.

   (c) (3p) Describe the Miller–Rabin test and how it can be used to help generate some of the public parameters involved in RSA. If your description involves randomisation, justify why the probability of success is high.

Solution (a) —

(b) One needs to be able to compute eth roots modulo a product $n = pq$ of two primes. The best approaches to this rely on factoring n. We studied the quadratic sieve for factoring n; this has expected running time about

$$O\left(e^{\sqrt{(\log n)(\log\log n)}}\right),$$

which means that it is subexponential.

(c) Bookwork for the test description.

One can use this test in generating the large primes needed for RSA, simply by picking large integers at random and testing whether they are primes until one has found two (likely) primes. By the prime number theorem, one is likely to succeed after not too many attempts, for the current sizes of primes needed for RSA, since roughly a proportion $1/\log N$ of the integers below N are prime.

The Miller–Rabin test itself is also likely to successfully identify primes, relying on the fact that at least 75% of numbers less than an odd composite number n are Miller–Rabin witnesses for the compositeness of n.

8. (a) (7p) Let $p = 503$ (a prime), $g = 53$ and $h = 204$. The element g is a primitive root modulo p. Solve $g^x \equiv h \pmod{p}$ using index calculus.

   The fact that $hg^{-88} \equiv 60 \pmod{p}$ and the following table might be helpful.

i	$g^i \pmod p$
457	$3^4 \cdot 5$
434	$3^3 \cdot 7$
136	$3^2 \cdot 7$
12	$2 \cdot 3^3$

(b) (1p) Say briefly how a table such as the above might be found in practice, and in particular why one is likely to have success in doing so.

Solution (a) Since $hg^{-88} \equiv 60 \equiv 2^2 \cdot 3 \cdot 5 \pmod p$, we have that

$$\log_g(h) \equiv 88 + 2\log_g(2) + \log_g(3) + \log_g(5) \pmod{(p-1)}.$$

We shall use the information in the table to compute these discrete logarithms. All congruences below are mod $p - 1 = 502$. First, the table gives

$$\begin{aligned} 457 &\equiv 4\log_g(3) + \log_g(5) \\ 434 &\equiv 3\log_g(3) + \log_g(7) \\ 136 &\equiv 2\log_g(3) + \log_g(7) \\ 12 &\equiv \log_g(2) + 3\log_g(3). \end{aligned}$$

Subtracting the third congruence from the first, we find

$$\log_g(3) = 434 - 136 \equiv 298.$$

Solving for $\log_g(2)$ and $\log_g(5)$, we then obtain

$$\log_g(2) \equiv 122 \quad \text{and} \quad \log_g(5) \equiv 269.$$

Thus, returning to the top computation,

$$\log_g(h) \equiv 88 + 2 \cdot 122 + 298 + 269 \equiv 397.$$

The (unique) solution (mod 502) is then $x = 397$.

(b) One can generate such a table by picking the exponent i at random mod $(p-1)$ and testing whether g^i is B-smooth for some small parameter B. Since there are quite a lot of smooth numbers, one is likely to hit these relatively often.

9. (a) (2p) Name two different ways in which elliptic curves are relevant to modern cryptography.

(b) (1p) Consider the elliptic curve over $\mathbb{F}_p$ ($p \geq 5$) given by

$$E : y^2 = x^3 + 3.$$

What does the Hasse bound say about the number of points of $E(\mathbb{F}_p)$?

(c) (2p) With E as above, how many points does $E(\mathbb{F}_7)$ have? How does this compare to the bound from the previous part?

(d) (3p) Describe how elliptic curve Diffie–Hellman key exchange works. Explain why it is not truly necessary for Alice and Bob to transmit the full pair (x, y) defining a point at each step, and why not doing so could be advantageous. (You may assume that we are working over a prime field $\mathbb{F}_p$ with $p \equiv 3 \pmod 4$.)

Solution (a) One way is for constructing seemingly secure cryptosystems, for example based on the DLP over an elliptic curve over $\mathbb{F}_p$, for which no algorithms performing better than $O(\sqrt{p})$ are known. Another is in factorisation: one can use Lenstra's elliptic curve factorisation method to factor fairly large integers.

(b) The discriminant is non-zero, and so the Hasse bound says that the number of points of $E(\mathbb{F}_p)$ is $p+1+t_p$ where $|t_p| \leq 2\sqrt{p}$.

(c) We draw up a table, using the fact that $y^2 = (7-y)^2$, meaning that the squares in $\mathbb{F}_7$ are $0, 1, 4, 2$, and that $(7-x)^3 = -x^3$:

x	x^3	x^3+3	y
0	0	3	–
1	1	4	± 2
2	1	4	± 2
3	-1	2	± 3
4	1	4	± 2
5	-1	2	± 3
6	-1	2	± 3

Taking into account the point at infinity O, there are thus $1 + 2 \cdot 6 = 13$ points in $E(\mathbb{F}_7)$. The previous bound implies that

$$|E(\mathbb{F}_7)| \leq 8 + 2\sqrt{7} < 14,$$

and is thus sharp for this curve.

(d) Bookwork for the first part.

It is not truly necessary for Alice and Bob to transmit the full pair (x, y) at each step, since y can be found fairly quickly from x using the curve's definition, at least up to a sign. One could thus transmit only x and one extra bit to indicate the sign of y, thus reducing the amount of data needing to be transmitted (roughly cutting it in half on average). (Note: it is quick to compute square roots mod p if $p \equiv 3 \pmod 4$, by fast powering.)

10. (a) (2p) Describe briefly the relationship between Pollard's $p-1$ method and Lenstra's elliptic curve factorisation method. (You do not have to describe in detail how either one works.)

(b) (2p) What is the expected running time of Lenstra's elliptic curve factorisation method in factorising N, if $N = pq$ is a product of two primes and $p < q$? Under what circumstances might this offer an advantage over the other factorisation methods of the course?

(c) (4p) This question is about finding a factor of the number $N = 2021 = 43 \cdot 47$ using Lenstra's method. Consider the equation

$$E : y^2 = x^3 + 3x - 3$$

defining an elliptic curve over various fields. The curve contains the point $P = (1, 1)$ over any field, and over $\mathbb{Z}/N\mathbb{Z}$. It turns out that

$$11P = \mathcal{O} \text{ over } \mathbb{F}_{43},$$
$$37P = \mathcal{O} \text{ over } \mathbb{F}_{47},$$

and that 11 and 37 are the smallest such numbers.

Describe how Lenstra's algorithm will run, motivating why it will find a non-trivial factor of N and stating after how many steps.

Solution (a) These algorithms work in the same way, except that Pollard's algorithm works in the setting of multiplication over $\mathbb{F}_p^\times$ whereas Lenstra's elliptic curve factorisation method works with addition on an elliptic curve over $\mathbb{Z}/N\mathbb{Z}$.

(b) Its average running time is approximately $O\left(e^{\sqrt{2(\log p)(\log\log p)}}\right)$. This would be better than, say, the quadratic sieve method for factoring if N has a relatively small prime factor.

(c) The algorithm will start with P and successively compute, over $\mathbb{Z}/N\mathbb{Z}$, the points

$$2!P = 2P, \quad 3!P = 3(2P), \quad 4!P = 4(3!P), \ \ldots$$

until it reaches $11!P$ (after 10–11 steps), where it will have tried to invert a number d that is not invertible in $\mathbb{Z}/N\mathbb{Z}$. Since $11P = \mathcal{O}$ over $\mathbb{F}_{43}$, this number will contain a factor 43, but since $11P \neq \mathcal{O}$ over $\mathbb{F}_{47}$, it will not contain a factor 47. Thus, by computing $\gcd(d, N) = 43$ using the Extended Euclidean algorithm, we will have found the factor 43 of N.