KTH Computer Science
and Communication

Homework I, Foundations of Cryptography 2017

Before you start:

1. The deadlines in this course are strict and stated on the course homepage.
2. Read the detailed homework rules at the course homepage.
3. Read about I and T-points and grades in the course description.

The problems are given in no particular order. If something seems wrong, then visit the course homepage to see if any errata was posted. If this does not help, then email `dog@kth.se`. Don't forget to prefix your email subject with `Krypto17`. We may publish hints on the homepage as if a problem appears to be harder than expected.

1 (2T) Describe the structure of a proof by reduction in cryptography as explained in class, i.e., describe the roles of definitions, assumptions, reductions, parties, adversaries, and conclusions. A fellow student that does not follow the course should be able to understand your description. You may show your description to somebody that does not follow the course to check this! Feel free to use handmade illustrations.

2 (2T) The standard definition of an efficient algorithm in complexity theory is an algorithm with polynomial running time (if it is uniform or not does not matter in this problem). Why is this notion not satisfactory when constructing algorithms for optimization problems, but it is rarely unsatisfactory when modeling adversaries in cryptography?

3 (3T) Suppose that $f(\cdot)$ is not negligible. If there exist integers $c, n_0 > 0$ such that $f(n) > n^{-c}$ for all $n > n_0$, then prove this and otherwise give a counter example.

4 (2T) List the indices of the functions below that are negligible functions. For example, if you think $f_1(n)$ and $f_2(n)$ are negligible and no other, then your answer should simply be "$1, 2$".

$$f_1(n) = n^{-\log n} \quad f_2(n) = n^{-256} \quad f_3(n) = 2^{-n} \quad f_4(n) = n^{-128n} \quad f_5(n) = f_2(n) + f_4(n)$$

To get any points your answer must be completely correct, i.e., this is an all-or-nothing problem. You do not need to motivate your answer for this problem.

5 (10I) Implement the AES cipher. A detailed description is found on Kattis `https://kth.kattis.com/problems/oldkattis.aes`. Feel free to consult different sources on how to make an efficient implementation, but any borrowed ideas should be explained briefly in the solutions submitted on paper. You must also be prepared to explain in detail what you did and why at the oral exam. Make sure that your code is commented and well structured. Up to 10I points may be subtracted if this is not the case.

6 In each case below, say as much as you can about the entropy of Y and motivate your answers. Make sure that you do not assume anything about the distribution of Y that is not stated explicitly. More precisely, for each description of the random variable Y given below, explain if, why, and how, the information given about Y:

1. is sufficient/insufficient to compute the entropy of Y,
2. allows you to give a closed expression of the entropy of Y, or
3. only allows you to bound the entropy of Y from above and/or below.

(Possibly in terms of the entropies of X and S.)

6a (1T) Let $Y = (X_1, \ldots, X_n)$ be a random variable over $\{0,1\}^n$ such that $\Pr[X_i = 1] = 1/2$ for $i = 1, \ldots, n$.

6b (1T) Let S be a uniformly distributed random variable over $\{0,1\}^{128}$ and define Y to be the top 64 bits of $\mathsf{AES}_k(S)$, where AES_k denotes the AES function for a fixed key k.

6c (1T) Let S be a uniformly distributed random variable over $\{0,1\}^{128}$ and define $Y = \mathcal{RO}(S)$, where $\mathcal{RO} : \{0,1\}^{256} \rightarrow \{0,1\}^{256}$ is a random oracle.

6d (1T) Let X, S, and T be independent random variables over $\mathbb{Z}_q$, where q is an odd prime, and define $Y = (X, S, S + T, X + T)$, where sums are taken modulo q.

6e (2T) Let $Y = (X_0, \ldots, X_n)$ be a random variable over $\{0,1\}^n$ such that $X_0 = 1$ and for every $(x_1, \ldots, x_{i-1}) \in \{0,1\}^{i-1}$ we have $\Pr[X_i = X_{i-1} | (X_1, \ldots, X_{i-1}) = (x_1, \ldots, x_{i-1})] = 2^{-i}$ for $i = 1, \ldots, n$.

6f (2T) Let f be a function, let X be a random variable over a set $\mathcal{X}$, and define $Y = f(X)$. Only the probability function $p_X(x)$ of X is given, not the one for Y.

6g (2T) Let f be a function, let Y be a random variable over a set $\mathcal{Y}$, and define $X = f(Y)$. Only the probability function $p_X(x)$ of X is given, not the one for Y.

7 (6T) Read the paper of Khazaei and Ahmadi on the Hill cipher. Describe in detail what kinds of attacks you can mount on the Hill cipher. Is it a known-plaintext, chosen-plaintext, etc attack? How many ciphertexts do you need? What is the approximate time complexity of the attacks?

8 Search for information about uniform and non-uniform adversaries.

8a (1T) Describe the difference in your own words.

8b (2T) Does it matter which view we take on efficient adversaries? (both in theory and practice) Are they equivalent?

9 Let $E_t : \{0,1\}^n \times \{0,1\}^{tn} \leftarrow \{0,1\}^n$ be an n-bit block cipher with tn-bit keys, consisting of a t-round Feistel network. Let "$\|$" denote concatenation and let f_i be the ith Feistel function. Then denote the key by $k = k_1\|k_2\|..\|k_t$, the plaintext by $L_0\|R_0 \in \{0,1\}^n$, and the output in round $s \geq 1$ by $L_s\|R_s$, i.e., the output ciphertext is $L_t\|R_t$. Assume that $f_i(k_i, \cdot)$ is pseudo-random function for a random k_i.

9a (2T) Show that if $t = 1$, then the Feistel network is not a pseudorandom permutation.

9b (4T) Show that if $t = 2$, then the Feistel network is not a pseudorandom permutation.

9c (10T) Show that if $t = 3$, then the Feistel network is not a pseudorandom permutation. (Hint: Look at several related inputs and outputs. Evaluate the permutation as well as its inverse on these.)

10 (3I) Implement modular exponentiation from modular multiplication. A detailed description is found on Kattis. `https://kth.kattis.com/problems/kth.krypto.modexp`. Use a big integer library for multiplication, e.g., GMP in C/C++ or `java.math.BigInteger` in Java. Make sure that your code is commented and well structured. Up to 3I points may be subtracted if this is not the case. Keep in mind that you must be able to explain your solution during the oral exam.

11 (3I) Implement Chinese remaindering. A detailed description is found on Kattis. `https://kth.kattis.com/problems/kth.krypto.crt`. Make sure that your code is commented and well structured. Up to 3I points may be subtracted if this is not the case. Keep in mind that you must be able to explain your solution during the oral exam.

12 (5I) Compute the factorization of an RSA modulus from its encryption and decryption exponents. A detailed description is found on Kattis. `https://kth.kattis.com/problems/kth.krypto.rsafact`. Make sure that your code is commented and well structured. Up to 5I points may be subtracted if this is not the case. Keep in mind that you must be able to explain your solution during the oral exam.

13 (2I) Determine basic properties of elliptic curves. A detailed description is found on Kattis. `https://kth.kattis.com/problems/kth.krypto.ellipticcurvepoints`. Make sure that your code is commented and well structured. Up to 2I points may be subtracted if this is not the case. Keep in mind that you must be able to explain your solution during the oral exam.

14 (4T) In some applications side channel attacks are a concern. Describe what a side channel attack is. Find as many side channels as possible that as been exploited in the research literature. Cite each relevant paper properly in your answer with a brief description in a few sentences.

15 The goal of this problem is to study the OpenSSL source code to get feeling for what real world code for cryptography can look like.

15a (1T) Identify and report the path to the file containing the optimized implementation of P-256.

15b (1T) Determine if any security critical bugs have been fixed in this code since it was committed to the code base.

15c (3T) Describe the techniques used in the implementation of P-256 to counter side channel attacks.

16 The generic elliptic curves covered in class uses separate code for doubling, adding, and treatment of the point at infinity. For some curves this is not necessary, i.e., there is no need for special code.

16a (3T) Dan Bernstein has published several papers about this. Read enough to be able to explain the key ideas.

16b (2T) What are the advantages of such curves in practice?

17 The goal of this problem is to prove the following implications covered in class. In other words, the difficulty in solving this problem is not understanding that the implications hold, but to write down a rigorous proof. Thus, in this particular problem, any handwaving give zero points. A proof consists of a description of an efficient reduction and a mathematical analysis thereof.

17a (2T) Prove that the DH assumption implies the DL assumption.

17b (2T) Prove that the DDH assumption implies the DH assumption.

18 Let $p = kq + 1$ and q be primes such that $\log q = n$, $\log k = n$ and such that the bit size of every prime factor of k is bounded by $\log n$. Let g be a generator of the unique subgroup of $\mathbb{Z}_p^*$ of order q. I pick $x \in \mathbb{Z}_q$ randomly and hand you $y = g^x$. Then you may ask me any number of questions of the form $u \in \mathbb{Z}_p^*$, which I answer by $u^x \bmod p$.

18a (2T) Explain how you can compute x efficiently (describe your algorithm and analyze its running time).

18b (1T) What is the important lesson to learn from this example? (This was mentioned in class, but you will not find it on any slides.)

18c (1T) How would you address this problem in an implementation of the protocol?

19 The goal of this problem is that you write out the details of a proof in cryptography on your own. We have already covered this result in class. Let $\mathsf{CS} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a public key cryptosystem. More precisely:

- Gen is a probabilistic key generation algorithm that on input 1^n (security parameter n in unary representation) outputs a key pair. We denote this by $(\mathrm{pk}, \mathrm{sk}) = \mathsf{Gen}(1^n)$.
- Enc is an encryption algorithm that takes a public key pk, a message $m \in \{0,1\}^n$, and randomness $r \in \{0,1\}^n$ as input and produces a ciphertext. We denote this by $\mathsf{Enc}_{\mathrm{pk}}(m, r)$.
- Dec is a decryption algorithm that takes a secret key sk and a ciphertext c as input and outputs the plaintext. We denote this by $m = \mathsf{Dec}_{\mathrm{sk}}(c)$.

Denote by $\mathsf{CS}^k = (\mathsf{Gen}^k, \mathsf{Enc}^k, \mathsf{Dec}^k)$ the cryptosystem defined as follows:

- Gen^k is identical to Gen
- Enc^k takes a public key pk, a message $m \in \{0,1\}^{n\times k}$, and randomness $r \in \{0,1\}^{n\times k}$ as input and outputs $(\mathsf{Enc}_{\mathrm{pk}}(m_1, r_1), \ldots, \mathsf{Enc}_{\mathrm{pk}}(m_k, r_k))$.
- Dec^k takes a secret key sk and a ciphertext $c = (c_1, \ldots, c_k)$ as input and outputs a plaintext $(\mathsf{Dec}_{\mathrm{sk}}(c_1), \ldots, \mathsf{Dec}_{\mathrm{sk}}(c_k))$.

19a (7T) Prove that if CS is secure, then CS^2 is secure.

19b (3T) Prove that for every polynomial $k(n)$, if CS is secure, then $\mathsf{CS}^{k(n)}$ is secure.

You need to be more rigorous than what we did in class that! Imagine that your life depended on convincing your worst enemy.