KTH Computer Science and Communication

Homework III, Foundations of Cryptography 2016

Before you start:

1. The deadlines in this course are strict. This homework set is due as specified at `https://www.kth.se/social/course/DD2448/subgroup/vt-2016-krypto16/page/deadlines-16`.
2. Read the detailed homework rules at `https://www.kth.se/social/files/5686fcd8f276542387729c18/solution_rules.pdf`.
3. Read about I and T-points, and how these translate into grades, in the course description at `https://www.kth.se/social/files/5692df7bf2765405aca1825f/course_description.pdf`.
4. You may only submit solutions for a nominal value of 50 points in total (summing I and T points). The total number of points below may be larger and this should be interpreted as giving you a way to choose problems you like.

The problems are given in no particular order. If something seems wrong, then visit `https://www.kth.se/social/course/DD2448/subgroup/vt-2016-krypto16/page/handouts-10` 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 `Krypto16`.

We may publish hints on the homepage as well if a problem appears to be harder than expected.

Definition 1 *The **RSA** assumption states that if $N = pq$, where p and q are randomly chosen primes with the same number of bits, $e \in \mathbb{Z}^*_{\phi(N)}$, and g is randomly chosen in $\mathbb{Z}^*_N$, then for every polynomial time algorithm A, $\Pr[A(N, e, g) = \beta \wedge \beta^e = g \bmod N]$ is negligible.*

Definition 2 *The **Strong RSA** assumption states that if $N = pq$, where p and q are randomly chosen primes with the same number of bits and g is randomly chosen in $\mathbb{Z}^*_N$, then for every polynomial time algorithm A, $\Pr[A(N, g) = (e, \beta) \wedge \beta^e = g \bmod N \wedge e > 1]$ is negligible.*

1 *This is probably a difficult problem, so for this particular problem (not for the rest of the homework) you may cooperate in any way within your study group (of three people) and you can simply copy the LaTeXsource from your joint solution your submitted solution.*

In class we considered the RSA signature scheme, i.e., RSA with full domain hash. In this problem we develop a different scheme based on the strong RSA assumption. Our construction is similar to some efficient *provably secure* signature schemes, but we only consider a simplified scheme and analyze its security in the random oracle model.

The private key of our scheme consists of two random $n/2$-bit safe[1] primes p and q. The public key consists of the modulus $N = pq$ and a random element g from the subgroup QR_N of quadratic residues in $\mathbb{Z}_N$. Suppose that $H : \{0,1\}^* \to \mathcal{P} \cap \{0,1\}^{n/3}$ is a random oracle, where $\mathcal{P}$ denotes the set of odd primes. A signature s of a message m is computed as $s = g^{1/H(m)} \bmod N$, where $1/H(m)$ should be understood as $H(m)^{-1} \bmod \frac{1}{2}(p-1)(q-1)$. To verify a signature s, one simply checks that $s^{H(m)} \bmod N = g$.

[1] A prime p is safe if $(p-1)/2$ is prime as well.

1a (2T) For a standard RSA modulus we do not require that p and q are safe. Prove that this does not change the hardness of factoring N in any essential way. Hint: Use the prime number theorem to estimate heuristically the probability that a randomly chosen prime is safe by chance.

1b (1T) Prove that if the strong RSA assumption holds, then the standard RSA assumption holds. (The opposite direction is unknown.)

1c (1T) Prove that the signature scheme is correct, i.e., that $(g^{1/H(m)} \bmod N)^{H(m)} \bmod N = g$ for every message m.

1d (2T) Let $p_1, \ldots, p_k \in \mathcal{P} \cap \{0,1\}^{n/3}$ be primes and let $g' \in \mathsf{QR}_N$ be randomly chosen. Prove that if we define $g = (g')^{\prod_{i=1}^{k} p_i} \bmod N$, then g is randomly distributed in QR_N. Thus, given a random element g' we can construct another random element g of which we can take any p_ith root modulo N efficiently.

1e (1T) Suppose that there exists a polynomial time algorithm A such that for random keys $(pk, sk) = ((N, g), (p, q))$ and random H,

$$\Pr[A^{\mathsf{Sign}_{sk}(\cdot), H(\cdot)}(pk) = (m, s) \wedge \mathsf{Verify}_{pk}(m, s) = 1 \wedge \forall i : m_i \neq m] \geq \delta \ ,$$

where m_i is the ith query to the signature oracle $\mathsf{Sign}_{sk}(\cdot)$ and δ is non-negligible, i.e., A breaks the signature scheme. (In the literature the random oracle is often implicit. Here we make it explicit.)

Prove that without loss of generality we may assume that A never asks the same query twice and that it always evaluates the random oracle H on the message m of its output.

1f (3T) Use the above to prove that given a random RSA modulus N and a random element $g' \in \mathsf{QR}_N$ you can generate a public key $pk = (N, g)$ such that you can simulate (without the secret key sk) a signature oracle $\mathsf{Sign}'(\cdot)$ and a random oracle $H(\cdot)$ such that

$$\Pr[A^{\mathsf{Sign}'(\cdot), H(\cdot)}(pk) = (m, s) \wedge \mathsf{Verify}_{pk}(m, s) = 1 \wedge \forall i : m_i \neq m] \geq \delta - \epsilon \ ,$$

where m_i is the ith query to the "signature oracle" $\mathsf{Sign}'(\cdot)$ and ϵ is exponentially small.

1g (1T) Prove that if A has polynomial running time $T(n)$ and j is randomly chosen in $\{1, 2, \ldots, T(n)\}$, then

$$\begin{aligned}&\Pr[A^{\mathsf{Sign}'(\cdot), H(\cdot)}(pk) = (m, s) \wedge \mathsf{Verify}_{pk}(m, s) = 1 \wedge \forall i : m_i \neq m \wedge m = m'_j] \\ &\geq \delta / T(n) - \epsilon'\end{aligned}$$

where m_i is the ith query to the signature oracle and m'_j is the ith query to the random oracle, and ϵ' is exponentially small. Hint: Exploit that the distribution of j is independent of everything else.

1h (3T) Let N be an RSA modulus, let $g' \in \mathsf{QR}_N$, and define $g = (g')^{\prod_{i \neq j} p_i} \bmod N$. Prove that if (m, s) satisfies $\mathsf{Verify}_{pk}(m, s) = 1$ and $H(m) = p_j$, then we can find integers a and b such that $ap_j + b\prod_{i \neq j} p_i = 1$ and construct (β, ρ) such that $\beta^\rho \bmod N = g'$ and $\rho > 2$.

1i (2T) Use the above observatations to describe an algorithm A' that runs A as a subroutine and breaks the strong RSA assumption, i.e., A' takes an RSA modulus N and a random element $g' \in \mathsf{QR}_N$ as input and must use A to output (β, ρ) such that $\beta^\rho \bmod N = g'$ and $\rho > 2$.

1j (2T) Suppose that we only wish to sign a polynomial $h(n)$ number of distinct messages known in advance (we can think of the messages as the integers $1, \ldots, h(n)$). Can you modify the signature scheme for this setting and prove its security without the random oracle?

2 (2T) Let p and q be distinct odd primes greater than five such that $(p-1)/2$ and $(q-1)/2$ are prime and define $N = pq$. What is the order of the largest cyclic non-trivial proper subgroup of $\mathbb{Z}_N^*$?

3 We consider variations of signature schemes to illustrate the diversity of even simple notions as signatures. For each subproblem below, describe the notion in terms of the algorithms involved, what they do, the security definition, and the motivation for introducing the notion. Do not simply copy a definition from the literature, instead *explain it in your own words*, i.e., this problem is about concepts and not mathematics. Good questions include, but are not limited to: Who holds which secret keys? Do we need a trusted party to help the signers and verifiers? What extra features are provided? What, if anything, remains private? What repudiation properties do we get?

3a (2T) Blind signatures.

3b (2T) Ring signatures.

3c (2T) Group signatures. (How do they differ from ring signatures?)

Hint: Do not get stuck in the technical details of any single paper. Browse multiple papers to understand the key ideas.

4 Consider the hash function defined as follows. Let $N = pq$ where p and q are randomly chosen safe primes of the same bit-size, and let g be randomly chosen in $\mathbb{Z}_N^*$ with order $(p-1)(q-1)/4$. Then define $h_{N,g}(x) = g^x \bmod N$.

4a (2T) Prove that a multiple of $(p-1)(q-1)/4$ can be computed from a collision.

4b (2T) Use this fact to prove that the hash function is collision-resistant under the strong RSA assumption.

5 (2T) Read about Lamport's one-time signatures and explain why you can not use a Lamport signature key pair more than once in general.

6 (10I) Implement the arithmetic of an elliptic curve. A detailed description is found on Kattis. `https://kth.kattis.scrool.se/problems/ellipticcurvearithm`. Make sure that your code is commented and well structured. Up to 10I 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.

7 (5I) Implement the recovery phase of Feldman's verifiable secret sharing scheme. A detailed description is found on Kattis. `https://kth.kattis.scrool.se/problems/feldman`. 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.

8 (10I) Implement the SHA-256 hash function. A detailed description is found on Kattis. `https://kth.kattis.scrool.se/problems/sha256`. Feel free to read from 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.