Homework 4

Due date: Thursday , April 8nd

Question 1 (20 points)

Let $\pi(N) = |\{p \leq N : p \text{ is prime }\}|$ denote the number of primes p between 2 and N. Suppose that we pick n-bit primes $p_1, \ldots, p_m$ and $q_1, \ldots, q_m$ and use these primes to generate m RSA keys $(N_i = p_i q_i, e_i, d_i)$ where $e_i d_i = 1 \mod \phi(N_i)$.

- **Part A.** Suppose that $N_1 \neq N_2$ and $\gcd(N_1, N_2) > 1$. Explain how to recover d_1 and d_2 from e_1 and e_2.
- **Part B.** Suppose that each prime p_i (resp. q_i) is picked uniformly at random from the set$\{p < 2^n :$ p is prime$\}$. What is the probability that $\gcd(N_1, N_2) \neq 1$? Express your answer using the prime counting function $\pi(.)$.
- **Part C.** Upper bound the probability that there exists $i < j$ such that $\gcd(N_i, N_j) > 1$. Express your answer using $\pi(.)$ and simplify your answer under the assumption that $\pi(x) = x/\ln x$. What value do you get when $n = 1000$ and $m = 2^{32}$?
- **Part D.** Suppose that our implementation of the RSA Key Generation algorithm is broken due to the use of a weak PRG. Instead of outputting a random prime in $\{p < 2^n : p \text{ is prime}\}$, our algorithm outputs a uniformly random prime from a subset $S \subseteq \{p < 2^n : p \text{ is prime}\}$ of size $t \ll 2^n$. How many of the RSA keys can we break in expectation?

  (**Hint**: Let $X_i = 1$ if and only if for some $j \neq i$ we have $N_i \neq N_j$ and $\gcd(N_i, N_j) \neq 1$. Your attack should run in time $O(m^2 \cdot \mathsf{poly}(n))$.)
- **Part E.** What value do you get from part D when $t = 2^{32}$ and $m = 2^{20}$?

Answer: ...

Resource and Collaborator Statement: ...

Question 2 (20 points)

Suppose we have 3 RSA public keys with (N_i, e_i, d_i) with $e_i = 3$ and a message $m < \min N_1, N_2, N_3$. Let $c_i = m^{e_i} \mod N_i$ for $i <= 3$. Show how to recover m. (**Hint:** You can use the Chinese Remainder Theorem).

You have two options to answer this question. Option 1 submit a typed solution (pseudocode) describing your attack. For full credit make sure that your solution handles all possible cases and provide a clear explanation of why your attack works. The second option

is to submit code on Gradescope. If you pick this option your question will be autograded i.e., you will not lose points for imprecise explanations as long as you pass all unit tests.

If you select to use the auto gradable option, your task will be to submit a file called "`Hw4Q2attack.py`" which includes the function `Recover_Message`$(\cdot)$. Your task is to complete this function. The initial template of this file is provided for your usage. Once you complete the function our designed test files on gradescope (in the autogradable platform) will call your completed function to recover the actual plaintexts. We have four total cases such that two of them are visible and the other ones are invisible until deadline due date. For full credit you need to pass all these four test cases.

Besides the template that we provided, you are also given another file which includes our challenge ciphertexts and their corresponding public keys. More specifically, we selected several messages $m_i < \min\{N_1^i, N_2^i, N_3^i\}$ in which N_1^i, N_2^i, N_3^i are the corresponding public keys. For each m_i we compute the corresponding ciphertexts $c_j^i = m_i^e \mod N_j$ for all $j \in \{1, 2, 3\}$ where $e = 3$. Several hardcoded examples are given in the file "`RSA_msg_recover.py`" which you can use to test your attack.

Answer: (Only required if you are submitting a written solution)

Resource and Collaborator Statement: ...

Question 3 (20 points)

Fix $N \in \mathbb{N}$ such that $N, e > 1$ and $\texttt{gcd}(e, \phi(N)) = 1$. Assume that there is an adversary $\mathcal{A}$ running in time t such that

$$\Pr\left[\mathcal{A}\left([x^e \mod N]\right) = x\right] \geq 0.01$$

where the probability is taken over the uniform choice of $x \in \mathbb{Z}_N^*$. Show how to construct an adversary $\mathcal{A}'$ with running time $t' = O\left(poly\left(t, \log_2 N\right)\right)$ such that

$$\Pr\left[\mathcal{A}\left([x^e \mod N]\right) = x\right] \geq 0.99 \ .$$

Hint: Use the fact that $y^{1/e} \cdot r = (y \cdot r^e)^{1/e} \mod N$. Here, $y^{1/e} = y^d \in \mathbb{Z}_N^*$ where d is a (secret) number such that $ed \equiv 1 \mod \phi(N)$. Also use the fact that, given $r \in \mathbb{Z}_N^*$, we can find a number r^{-1} such that $rr^{-1} = 1 \mod N$.

Answer:

Resource and Collaborator Statement: ...

Question 4 (20 points)

Let $pk = (N, e) \in \mathbb{N}$ such that $N, e \geq 1$ and $\texttt{gcd}(e, \phi(N)) = 1$ be the public-key for an RSA encryption scheme and let $n = \lceil \log_{256}(N) \rceil$. To convert a bit string $x = x_1, \ldots, x_t \in \{0,1\}^t$ with to an integer in $\mathbb{Z}_N$ we define $\textbf{Int}(x) = \sum_{i=1}^{t} x_i 2^{t-i}$.

Let m be the message "Pay Alice the following amount from Bob's bank account (USD): 50" and let μ denote a customized character to byte mapping such that $\mu(0) = 0^8, \mu(1) = 0^71, \ldots, \mu(9) = 00001001$, while other characters like 'a' or 'B' are mapped to bytes outside the range $[00000000, 00001001]$. Given $m = c_1, \ldots, c_{|m|}$ let $\mu(m) = \mu(c_1)\| \ldots \|\mu(c_{|m|})$ denote an encoding of m in bits (here $|m|$ denotes the number of characters in the message m). Similarly, given a bit string $x = x_1, \ldots, x_t \in \{0,1\}^t$ let $\mathbf{Int}(x) = \sum_{i=1}^{t} x_i 2^{t-i}$. Suppose that Bob sends Alice $\sigma = \mathbf{Sign}_{sk}(m) = \mathbf{Int}(\mu(m))^d \mod N$. Show how Alice can produce a signature σ' for a message authorizing the bank to transfer more than \$50 (you may assume that $|m| < n - 20$). Alice want to claim as much money as possible, but knows that Bob has \$750 million in his bank account and she will only be able to submit one signature σ' to the bank.

You may assume that $e \in \{3, 5, 7\}$. (Hint: The amount of money you can feasibly claim might depend on the public key e)

This question is a coding problem and you have to submit your Python code on Gradescope. We provide a template for you as well as some auxiliary files to conduct you attack. The template file is `Hw4Q4attack.py` and the auxiliary is `Hw4Q4HelperFunctions.py` and you have to modify the template to complete the function for attack i.e., `Forge_CS555` and for submission you should submit just `Hw4Q4attack.py` as your solution on Gradescope.

The auxiliary file `Hw4Q4HelperFunctions.py` describes some useful functions like the signing, verification, mentioned encoding schemes and decoding functions which may be used during your attempt to break the security of the described signature scheme. Your tasks is to complete the signature forgery function `Forge_CS555`. You will be given a signature σ of the original message for \$50 as well as the public key. You need to output a forged σ' authorizing a larger amount x, but subject to the constraint that $x \leq$ \$750 million. To pass each unit test you will need to ensure that you find the biggest value of x which Alice can feasibly forge. All these test cases remain invisible until the due date, however, you can test if your solution pass the test or not by receiving its corresponding points for the passed one.

Answer: (Submit code on Gradescope)

Resource and Collaborator Statement: ...

Question 5 (20 points)

Consider the following public key encryption scheme based on RSA. Let $pk = (N, e)$, $sk = (N, d)$ and let $G : \mathbb{Z}_N \to \{0,1\}^\lambda$ be random oracles. Let $\mathtt{Enc}_{pk}(m; r) = (r^e \mod N, G(r) \oplus m)$ where $r \in \{0,1\}^\lambda$ is picked uniformly at random. We also define $\mathtt{Dec}_{sk}(c = (s,t)) = t \oplus G(s^d \mod N)$. Assuming that the RSA-inversion assumption holds. Is this scheme CPA-Secure, CCA-Secure or insecure? If your solution is insecure you should show how a CPA-Attacker can win the security experiment. If your answer is CCA-Secure you should prove it. If your answer is CPA-Secure you should prove the solution is CPA-Secure and show how an attacker wins the CCA-security game.

Answer:

Resource and Collaborator Statement: ...