Problem Set 4

This problem set is due on *Monday, April 11, 2016* at **11:59 PM**. Please note our late submission penalty policy in the course information handout. Please submit your problem set, in PDF format, on Gradescope. *Each problem should be in a separate PDF.* Have **one and only one group member** submit the finished problem writeups. Please title each PDF with the Kerberos of your group members as well as the problem set number and problem number (i.e. *kerberos1_kerberos2_kerberos3_pset3_problem1.pdf*).

You are to work on this problem set with groups of your choosing of size three or four. If you need help finding a group, try posting on Piazza or email `6.857-tas@mit.edu`. You don't have to tell us your group members, just make sure you indicate them on Gradescope. Be sure that all group members can explain the solutions. See Handout 1 (*Course Information*) for our policy on collaboration.

Homework must be submitted electronically! Each problem answer must be provided as a separate pdf. Mark the top of each page with your group member names, the course number (6.857), the problem set number and question, and the date. We have provided templates for LaTeX and Microsoft Word on the course website (see the *Resources* page).

Grading: All problems are worth 10 points.

With the authors' permission, we may distribute our favorite solution to each problem as the "official" solution—this is your chance to become famous! If you do not wish for your homework to be used as an official solution, or if you wish that it only be used anonymously, please note this in your profile on your homework submission.

Our department is collecting statistics on how much time students are spending on psets, etc. For each problem, please give your estimate of the number of person-hours your team spent on that problem.

Problem 4-1. Quadratic Residues modulo Composite N

Let $N = pq$ be the product of two distinct odd primes.

(a) How many square roots does each element $x \in \mathbb{Z}_N^*$ have? Justify your answer with Chinese Remainder Theorem.

(b) What if $N = pqr$ be the product of three distinct odd primes?

(c) You are given an integer N and told it is either a product of two or three distinct odd primes. In addition, you have a magic box that can tell apart squares from nonsquares modulo this N. You can assume that the box is always correct. How would you use this box to determine whether N has two or three prime factors?

(d) You are given an integer $N = pq$ be the product of two distinct odd primes. In addition, you have a magic box that, on square $y \in \mathbb{Z}_N^*$, outputs $x \in \mathbb{Z}_N^*$ such that $x^2 = y \pmod N$. You can assume that the box is always correct. How would you use this box to factor N?

Problem 4-2. Attacking the Fourth Dimension

One of the more subtle threats to the security of cryptographic systems is the side-channel attack. Side-channel attacks utilize information about the execution of cryptographic operations, other than the inputs and outputs, such as execution time, power consumption, cache hits and misses, temperature, and even sound. David Brumley and Dan Boneh's paper, *Remote Timing Attacks are Practical* ([BB05]), documents a successful key recovery attack on one of the most universally used cryptography libraries, OpenSSL (https://www.openssl.org). Their attack operates on an OpenSSL version from 2003, exploiting the use of CRT (Chinese Remainder Theorem) and Montgomery modular arithmetic. We recommend reading all of [BB05] in your spare time, but Sections 2.2, 2.3, 3 and 3.1 will be the most important for solving this problem.

(a) Let R be the Montgomery scaling factor as in [BB05]. We denote the inverse of R modulo m as R_m^{-1}. [BB05]'s attack requires crafting a ciphertext u such that $g = uR \bmod q$ is very close to (but less than) q. One way would be to compute $gR_q^{-1} \bmod N$, but we cannot compute R_q^{-1} because q is secret. [BB05] suggests that simply using R_N^{-1} (where R and N are public) suffices. **Show that $R_N^{-1} * R \equiv 1 \bmod q$.**

(b) There are rumors that the 6.857 staff have not upgraded OpenSSL since 2003. Your task is to recover the factors of a 1024-bit secret key from the server using the attack detailed in [BB05]. The target server is located at http://6857rsa.csail.mit.edu:8080 and serves /gen_practice, /decrypt and /guess endpoints. The server sets $R = 2^{512}$ and returns the (simulated) time of decryption. Given any team name string, a new RSA key pair will be generated and stored for any future decryption requests. *On our server, the 496 most sigificant bits are recoverable through zero-one gap with neighborhood analysis, while the final 16 bits must be brute forced.* Our simulation uses a probability of extra computation of $\Pr[\mathsf{ExtraReduction}] = \frac{g \bmod q}{R}$ (twice that of in the paper), and exaggerates this extra time to make the attack achievable in less queries. Our simulation does not incorporate Karatsuba vs. normal multiplication methods.

The server will generate and reveal practice secret keys at the /gen_practice endpoint (detailed in the Python template).

Once you have recovered q for your team name, submit it to the /guess endpoint to receive credit for this part. We have provided a template in attack.py that includes API details.

Please also submit your code and an explanation of your strategy to Gradescope.

Problem 4-3. Elliptic Curves

Consider the following elliptic curve parameters:

- The elliptic curve equation is $y^2 = x^3 + 2x + 6$ (mod p)
- The prime number p=1021

In last week's recitation we covered calculations on elliptic curves. You can read Prof. Rivest's notes (pages 10-13) of the following link: http://courses.csail.mit.edu/6.857/2016/files/L13-groups-DH-key-exchange-elliptic-curves.pdf where the operations on elliptic curves are defined.

You can use any programming language environment to perform your computations. Specifically you may find SageMath useful (http://www.sagemath.org)

Let G=(644,772) be a generator for the elliptic curve group mentioned above.

(a) Calculate P=2G

(b) Find the order of G in the elliptic curve. Recall that the order of an element Q on the elliptic curve is the smallest positive integer k such that kQ=∞ where ∞ is the point at infinity defined in the lecture notes above.

(c) Consider the Pedersen commitment scheme for elliptic curves. In this scheme we are given a generator G of a group with prime order q. We are also given H where H=aG for secret a. Then, if we want to commit a value x (where $0 \leq x < q$) we simply commit C=xG+rH. Here r is chosen randomly by the sender ($0 \leq r < q$)).

Let G=(644,772) as above and let H=(937,302). Let the randomly selected r be equal to 865 for this problem. Commit the value x=345 according to the above scheme and return the commitment C for this x.