CS 4235 / 6035 Introduction to Information Security
Homework Two - Public Key Cryptography
Spring, 2016
Due at Start of Class on Thursday, 11 February

You may collaborate on this homework, but you must list your collaborators. If you use any resources, cite them! Please print this document and write your answers in the space provided. Show enough work that I can give partial credit.

Problem Zero (0 Points)

Your Name: ______________________________ Your GTID: ___________________________

Collaborators: ____

Problem One (2 Points)

In shared-key cryptography keys are generally 256 bits or less. Why are numbers in public-key cryptosystems typically on the order of thousands of bits long? (This question may require outside sources.)

Problem Two (2 Points)

Both digital signatures and message authentication codes provide for data integrity. Why don't message authentication codes provide *nonrepudiation*?

Problem Three (2 Points)

In our unit on access control we will learn that in some protocols a random digital signature challenge is sent for authenticatication purposes. For example, Alice may send Bob a random number to sign for Bob to prove his identity. With this in mind, why is it a bad idea to use the same RSA key pair for signing and encryption?

Problem Four (2 Points)

Recall that the RSA assumption is that, given $c = m^e \bmod n$, finding m given only c, e & n is hard (if we choose our key correctly). Suppose that you invent an algorithm, $D(c, e, n)$, that can find m for 1% of c. Give pseudocode for an algorithm that can use $D(c, e, n)$ to find m easily for any c. (Hint: Review the basic number theory that we discussed in class.)

Problem Five (2 Points)

Find 3^{123} mod 17 with only pen and this paper. Show each step! Put a box around your final answer. You may use a computer to double check your final answer.

Problem Six (3 Points)

Show that digital signatures under plain RSA are insecure. (Plain RSA means that signing is done by calculating m^d mod n, $m < n$, with no padding or hashing of m.) Write an algorithm that, given someone's public signature verification key, (e, n), can easily generate a (message, signature) pair for some message that the private key holder never intended to sign. For full credit your algorithm should not require a signing oracle and your algorithm should generate "signed" messages where the your message is different than its signature. (Hints: The message need not have any meaning to count as a forgery. Think backwards.)

Problem Seven (3 Points)

This problem shows the importance of picking *p* & *q* randomly when generating RSA keys. For this problem all variable names are as we used in class when we discussed the RSA encryption scheme. Suppose that an attacker can guess something about *p* & *q* other than *n* (= *pq*). Given a public key (*n*, *e*), a ciphertext, *c*, and some leaked information about *p* & *q,* can an attacker find the message, *m*?

Suppose that an attacker can guess that the distance between *p* & *q* is exactly 99,756 (q - p = 99,756). Given that and the following, find *m*. Show work and explain steps for full credit.

n=13407807929942597099574024998205846127479365820592393377723561443721764070199905996922954027860345430630722069842959996915503663917498816880633795704557141

e=65537

c=8005759326399040081769730394944621533886049862669982424066212838891731754896891137854460095179708123649729805240187601296323531253803144998960099456881193

For easy cutting and pasting, I have copied the above numbers to http://pastebin.com/cje8S1pz. Your answer may remind you of a famous mathematical constant. (Protip: For my calculations I used Python's `pow(a, b, c)` method, which returns a^b mod *c*. I used `PowerMod(...)` at http://www.wolframalpha.com/ to calculate a modular inverse, and I used Wolfram Alpha for a large square root.)

Problem Eight (4 Points)

In class we learned the Diffie-Hellman key exchange protocol for Alice and Bob. Suppose that Alice, Bob and Charlie need to share a common secret key. List the steps that would be involved in a secure three-way Diffie-Hellman key exchange. (Disregard man-in-the-middle attacks. Assume that Alice, Bob and Charlie all trust each other completely.)