PUBLIC-KEY CRYPTOGRAPHY

LAMC HIGH SCHOOL 1

1. How Public Keys Work, via KidRSA

Last week, we saw that there was such a thing as a perfect code: by using a truly random one-time pad cipher, no one could ever decipher your messages. The problem is, in order for a one-time pad to work, both the sender and recipient of the message need to have access to the key - and if you send the key to the recipient, that message will be vulnerable, because you can't encrypt a one-time pad without using up the same quantity of one-time pad as you're sending. To be able to send keys securely, the best thing would be an encryption scheme in which you can't deduce the decryption from the encryption - where the lock is different from the key. These are called Public-Key (or Asymmetric) ciphers.

The KidRSA Cryptography System: The KidRSA system first requires the recipient of the message generate two keys: one public, (e, n) (the e stands for encryption) and one private, (d, n) (the d stands for decryption) as follows:

(1) Choose four integers a, b, a', b' and set $M = ab - 1$
(2) Set $e = a'M + a$, $d = b'M + b$, and $n = (ed - 1)/M$
(3) (e, n) is the public key, displayed to everyone, and (d, n) is the private key, kept secret

Problem 1: Is n always an integer? Show your work.

To encrypt using KidRSA, your message needs to be a positive integer m both relatively prime to, and less than, n. The encrypted message is $m \cdot e \pmod{n}$. To decrypt a message, multiply it by d, then reduce modulo n.

Problem 2: Show that the decrypted message will always be m.

Problem 3: Encrypt, then decrypt the message "405" by starting with the numbers $a = 5, b = 3, a' = 7, b' = 5$.

Problem 4: Create a public and private key of your own; give your public key to your neighbor and have them send you a message.

The idea behind KidRSA is that, while it's very easy to multiply numbers, it seems hard to find the multiplicative inverse of one number mod another. Unfortunately for the secrecy of whatever messages you just sent, it isn't actually hard.

Problem 5: Show that if you can find integers r, s such that $r \cdot e + s \cdot n = 1$, then r is the multiplicative inverse of e mod n.

You can find numbers r, s as above by using the "Extended Euclidean Algorithm":

(1) Create three columns: in the first, put n, then e. In the second, put 1, then 0, and in the third, put 0, then 1.
(2) At each point, when we have rows $n-1$ and n, to get the $n+1$st row, look at the numbers in the left column. Subtract as many copies of the smaller from the larger as possible, while remaining positive, and write the result in the $n+1$st row. Then perform the same subtractions in the other columns to get the other entries of the $n+1$st row.
(3) Once there is a row where the left-hand entry is 1, the middle entry will be s, and the right entry will be r.

Problem 6: Explain why the Extended Euclidean Algorithm works to find r and s.

2. RSA

The RSA public-key encryption system is the current standard of encryption. Named after creators Ron Rivest, Adi Shamir, and Leonard Adelman, who described it in 1977, it was the first publicly known example of a secure, practical public-key encryption algorithm - before then, it wasn't even known that such a thing was possible.

KidRSA wasn't secure because the underlying problem, that of finding modular inverses, is relatively easy. The underlying problem of RSA is that of factoring, which to date, is still quite hard (and will likely one day be proven to be hard). Before describing it, we need one additional concept.

Definition: The **Euler totient function** ϕ, is defined on positive integers by setting $\phi(n)$ equal to the number of positive integers less than or equal to n which are relatively prime to n.

The number theory behind the totient function is deep and beautiful, but for understanding RSA all we need is that, when p and q are distinct primes, $\phi(pq) = (p-1)(q-1)$.

The RSA Encryption Scheme: Here is how to creat your own RSA keys:

(1) Choose distinct prime numbers p and q.
(2) Set $n = pq$, and compute $\phi(n) = (p-1)(q-1)$.
(3) Choose a number e relatively prime to $\phi(n)$, and set d equal to e^{-1} mod $\phi(n)$.
(4) Your **public key** is (e, n) and your **private key** is (d, n)
(5) To encrypt a message m, less than and relatively prime to n, you send the text $c = m^e$ mod n, and you decrypt this message by computing c^d mod n.

Problem 7: Send some messages using RSA with $p = 3$, $q = 11$, $e = 3$, and $d = 7$.

For RSA to be both secure and usable, it needs to meet our four criteria from last week. Let's examine each step, and make sure that the correct pieces are easy/hard.

(1) Finding large prime numbers (practically, RSA uses primes hundreds of digits long) is complicated, but it turns out, computationally easy, even though factoring is hard.
(2) Computing n, $\phi(n)$, and finding an e relatively prime to $\phi(n)$ are all very easy; finding d we know is easy from breaking KidRSA.
(3) The actual encryption and decryption can be done quickly via a method known as "fast modular exponentiation." To perform fast modular exponentiation of $m^e \bmod n$, square $m \bmod n$ repeatedly, until its exponent is the largest power of 2, less than e; call it k_0. Then square m repeatedly again, until its exponent is the largest power of 2 less than $e - k_0$, call it k_1. Eventually we'll know $m_0^k \bmod n$, $m_1^k \bmod n$, $m_2^k \bmod n$,..., and their product will be $m^e \bmod n$.

Problem 8: Show that RSA encryption works: that $(m^e)^d = m^{ed} = m \bmod n$. You may want to use Fermat's little Theorem, saying that whenever a and r are relatively prime, $a^{r-1} = 1 \bmod r$, and the fact that $ed = 1 \bmod \phi(n)$.

Problem 9: Suppose Alice and her friend Bob both have their own public and private RSA keys. How can Alice use those keys to send Bob a secure message, and have him be convinced that the message really came from Alice?

Problem 10: If $n = pq$ is the product of two distinct odd primes, show that at least one out of every three consecutive integers is relatively prime to n, and so if the last digit of a message is kept free, it is always possible to choose a final digit so that the message is relatively prime to n.

Problem 11: Show that it's important to keep $\phi(n)$ secret by finding formulas for p and q in terms of n and $\phi(n)$.