15-503/859P: Cryptography **Lecturer:** Manuel Blum
Topic: LALALA **Date:** 10 April 2006
Scribe: Yinmeng Zhang

21.1 Overview

1. A Nice Theorem
2. PKE with $f(x) = x^2 \pmod N$

We will use the notation $m \circ n$ to mean m concatenated with n.

21.2 A Nice Theorem

Theorem:

Let $N = p_1^{e_1} * \ldots p_k^{e_k}$ be an odd positive composite number (where the p_i are distinct primes). Let $a \in \mathbb{Z}_n^*$ be a square $\pmod N$.

If x, y are *distinct* roots of equation (1) – that is, $x \neq \pm y$, then $gcd(x + y, N)$ *splits* N – that is, $gcd(x + y, N)$ is a nontrivial factor of N (not 1 or N).

Notice that two distinct roots exist iff N is composite, so the ability to solve quadratic equations mod N implies the ability to factor N. Pairs x, y are the "gold rings" of factoring algorithms.

Proof:

$$x^2 \equiv a \equiv y^2 \pmod N$$

So

$$N | x^2 - y^2$$

or

$$N | (x + y)(x - y)$$

N cannot divide $x + y$ or $x - y$ separately, since this would imply $x \equiv \pm y \pmod N$, which contradicts the assumption that they are distinct. Therefore, $x + y$ has at least one but not all factors in common with N – thus $gcd(N, x + y)$ is a nontrivial factor of N. ■

This proof is so cute, it could just become the first question on the final.

21.3 PKE with $f(x) = x^2 \pmod N$

Today we discover that number theorists were secretly the great cryptographers of the last few centuries.

As usual, we take $N = P_1 * P_2$ the product of two distinct odd primes. Fix N_A as Alice's PUBLIC KEY, and its factorization P_{A1}, P_{A2} as her PRIVATE KEYS.

First stab at public-key encryption or PKE using $f(x) = x^2 \pmod N$:

For any message $m \in \mathbb{Z}_N^*$, we encrypt using

$$E(m) = m^2 \pmod N$$

and decrypt using

$$D(m^2) = \pm m_1, \pm m_2$$

21.3.1 The Pluses

1. *Fast!*

   Encryption is obviously polynomial time, but what about decryption? Alice should be the only one who is able to decrypt messages, and Alice knows what P_1 and P_2 are. If she can reduce the problem to finding the roots of $m^2 \pmod P)_1$ and $m^2 \pmod P)_2$, then she can apply the CRT to find the roots mod N. Finding roots is so much fun, it could just become the second question on the final.

2. *Hard as factoring to invert $E(m)$*

   If you have a Magic Box that on input "a" (a square) returns x s.t. $x^2 \equiv a \pmod N$, then you can factor N.

   A simple expected polynomial time algorithm is to repeatedly pick x uniformly at random in $\mathbb{Z}_n^*$, square it, and query the box. With 1/2 probability you will get a distinct root, and be able to factor N. This is an expected polynomial time algorithm.

   So, if can decrypt random numbers, then can factor N. Turning this on its head, we can encrypt random numbers! Good for sending random session keys.

21.3.2 The Minuses (with Fixes)

1. *We can only encrypt messages which are in $\mathbb{Z}_N^*$.*

   If we want to encrypt messages which are much bigger than N, we can send it $\log N$ bits at a time, reducing each piece to $\mathbb{Z}_N^+$. Sending messages which are not coprime with N is still a problem, since anyone could run Euclid's algorithm to retrieve the common factor.

2. *If an eavesdropper Eve knows that the message is one of a small class of messages (such as {yes, no, maybe}), then she can just encrypt each of the possible messages herself and compare against $E(m)$.*

   Notice that this would be a problem for *any* deterministic encryption scheme. To fix this, we will have to add randomness (see below).

3. *$D(E(m))$ is a set of four numbers. Which is the original m?*

   We might be tempted to say "the one that looks like English, duh", but unfortunately one of our pluses was that we could encrypt random numbers.

   To cut $D(E(x))$ down to two numbers, we can require messages to be in the left half of $\mathbb{Z}_n^*$; this will not change the distribution of encrypted messages. To get from two to one, we use the following clever

   Workaround (known padding): Before encrypting, pad m with a known prefix (or suffix) such as THISISTHEREALMESSAGE.

   This is not perfect. What if both messages have the prefix/suffix? What if you can use the known prefix/suffix to recover information?

4. *If you encrypt the same message twice with different keys, any eavesdropper who sees both encryptions can recover the message without knowing how to factor.*

   That is, suppose you are working with Alice and Bob with independent public keys N_A and N_B. If you send the message $E_{N_A}(m)$ to Alice and cc $E_{N_B}(m)$ to Bob, any Eve who intercepts both messages can find m as follows:

   Using CRT, calculate $m^2 \pmod{N)_1 N_2}$. Since $m < N_1$ and $m < N_2$, $m^2 < N_1 N_2$. So, $m^2 \pmod{N_1 N_2}$ is just plain m^2, which we can find the square root of using binary search. This is so devilish, it could just become third question on the final.

   Workaround (random padding): Before encrypting, pad m with a random r, and send the pair $(E(m \circ r), r)$.

   Notice how we can combine this with the previous workaround: since r is sent along with the message, we can use it to distinguish roots.

21.3.3 Improved PKE Scheme

Given $m \in \mathbb{Z}_n^*$,

To encrypt we choose a random r and send $((r \circ m)^2 \mod N_A, r)$,

To decrypt a pair (x^2, r), we find the square roots of x^2 and choose the one in the left half of $\mathbb{Z}_N^*$ which begins with the string r.