University of Maryland
CMSC456 — Introduction to Cryptography
Professor Jonathan Katz

Lecture 29

1 Public-Key Encryption Based on Quadratic Residuosity

Last time, we saw that an encryption scheme in which a "0" was represented as (say) a quadratic residue and a "1" was represented as a quadratic non-residue might be secure. Unfortunately, the "naive" scheme we gave last time did not seem to be implementable (since it was not clear how the sender would generate a random quadratic non-residue)! We now discuss some ways to get around this severe drawback.

We first note a fix which solves our problem (somewhat) but has drawbacks of its own.

1. $\mathcal{K}(1^k)$ generates two random primes p, q with $|p| = |q| = k$ and computes $N = pq$. The public key is N and the secret key is (p, q).

2. To encrypt "0", the sender chooses ℓ independently random quadratic residues in $\mathbb{Z}_N^*$; to encrypt "1", the sender chooses ℓ independently random elements with Jacobi symbol $+1$. The ciphertext consists of all ℓ elements.

3. Upon receiving ciphertext $C_1, \ldots, C_\ell$, the receiver (using p, q) determines whether either (1) each C_i is a quadratic residue or (2) at least one of the C_i is not a quadratic residue, and decrypts accordingly.

First note that this scheme is implementable (we discussed last time how the sender can pick random quadratic residues and how the sender can pick random elements with Jacobi symbol $+1$). Furthermore (the proof is somewhat involved and we will not give it here), the scheme is secure. But we can note two drawbacks of the scheme. First, the ciphertext length has been increased by a factor of ℓ and we are still encrypting only a single bit! Second, this scheme has a *small probability of decryption error*. In particular, when the sender encrypts a "1" then with probability $(\frac{1}{2})^\ell$ the sender chooses ℓ quadratic residues and sends them. But then the receiver will interpret this as a "0"! Of course, the effect this has in practice can be minimized by setting ℓ large enough. But then this severely reduces the efficiency (in terms of both computation and communication) of the scheme.

So the question remains: can we construct a scheme in which a *single* quadratic element (i.e., residue/non-residue) is transmitted, and which is implementable (and of course secure)? In fact, we can by using a neat "trick" whereby the recipient helps out the sender in the following way: Note that it is easy *for the receiver* to choose a random quadratic non-residue (since the receiver knows the factorization of N). The receiver can thus "help out" the sender by including a quadratic non-residue y as part of his public key. Now, the sender can choose a random quadratic residue as before (that is, by choosing random $x \in \mathbb{Z}_N^*$ and squaring it). But the sender can now also choose a random quadratic non-residue by choosing a random quadratic residue z and computing z *cdoty*. We give the details now.

1. $\mathcal{K}(1^k)$ generates two random primes p, q with $|p| = |q| = k$ and computes $N = pq$. Then, choose an arbitrary quadratic non-residue y with Jacobi symbol $+1$ (note that this can now be done efficiently since the factorization of N is known). The public key is (N, y) and the secret key is (p, q).

2. The sender chooses a random $x \in \mathbb{Z}_N^*$ and computes $z = x^2$. Note that z is a random quadratic residue. To encrypt "0", the sender sends $C = z$. To encrypt a "1", the sender sends $C = z \cdot y$. Note that in the first case, C is a random quadratic residue. In the second case C is a random quadratic non-residue with Jacobi symbol $+1$ (we leave verification of this fact to the reader).

3. Upon receiving ciphertext C, the receiver (using p, q) determines whether C is a quadratic residue or not, and decrypts accordingly.

It should be clear that the receiver always decrypts correctly.

We now discuss the security of the scheme. Before we can formally prove security, we must formally state our hardness assumption regarding deciding quadratic residuosity.

Definition 1 *We say deciding quadratic residuosity is (t, ϵ)-hard (relative to some algorithm $\mathcal{K}$ for generating moduli N) if, for all algorithms A running in time t we have:*

$$\left|\Pr[N \leftarrow \mathcal{K}; y \leftarrow \mathcal{QR}_N : A(N, y) = 1] - \Pr[N \leftarrow \mathcal{K}; y \leftarrow \mathcal{QRN}_N^{+1} : A(N, y) = 1]\right| \leq \epsilon,$$

where $\mathcal{QR}_N$ denotes the set of quadratic residues modulo N and $\mathcal{QRN}_N^{+1}$ denotes the set of quadratic non-residues in $\mathbb{Z}_N^*$ with Jacobi symbol $+1$.

We may now state and prove the security of the scheme:

Theorem 1 *Assuming that deciding quadratic residuosity (relative to $\mathcal{K}$) is (t, ϵ)-hard, the encryption scheme outlined above is $(t, 2\epsilon)$-secure against ciphertext-only attacks.*

Proof Recalling the definition of security against ciphertext-only attacks, we need to show that for any algorithm A running in time t we have:

$$\begin{aligned} 2\epsilon \;\geq\; & \left|\Pr[(N, y) \leftarrow \mathcal{K}; C \leftarrow \mathcal{E}_{N,y}(0) : A(N, y, C) = 0]\right. \\ & \left. - \Pr[(N, y) \leftarrow \mathcal{K}; C \leftarrow \mathcal{E}_{N,y}(1) : A(N, y, C) = 0]\right|. \end{aligned}$$

For simplicity, we use the following (equivalent[1]) formulation:

$$\mathsf{Adv}_A \stackrel{\text{def}}{=} \left|2 \cdot \Pr[(N, y) \leftarrow \mathcal{K}; b \leftarrow \{0,1\}; C \leftarrow \mathcal{E}_{N,y}(b) : A(N, y, C) = b] - 1\right| \leq 2\epsilon.$$

We assume toward a contradiction that this is *not* true; i.e., that there exists an algorithm A running in time t for which $\mathsf{Adv}_A > 2\epsilon$. We will then use this algorithm to construct an algorithm A' deciding quadratic residuosity with probability better than ϵ, a contradiction.

Consider the following algorithm A' which gets as input a modulus N and an element $y \in \mathcal{J}_N^{+1}$ (which is either a quadratic residue or not):

[1] Equivalence was proved in class, and is straightforward.

$A'(N, y)$
$b \leftarrow \{0, 1\}$
choose $x \leftarrow \mathbb{Z}_N^*$; set $z = x^2$
if $b = 0$ set $C = z$
if $b = 1$ set $C = zy$
run $A(N, y, C)$ to get output b'
if $b = b'$ output 1 else output 0

We are interested in the difference between the probability that A' outputs 1 when y is a quadratic residue and the probability that A' outputs 1 when y is a quadratic non-residue (cf. Definition 1).

Let us first examine what happens when y is a quadratic non-residue (with Jacobi symbol $+1$). In this case, note that A' exactly simulates the encryption scheme for A; in other words, the view of A is the same as what it would see if a random public key were generated and then a random bit encrypted. Since A' outputs 1 exactly when A correctly guesses the value of the encrypted bit, we have:

$$\begin{aligned} &\Pr[N \leftarrow \mathcal{K}; y \leftarrow \mathcal{QRN}_N^{+1} : A'(N, y) = 1] \\ &= \Pr[(N, y) \leftarrow \mathcal{K}; b \leftarrow \{0, 1\}; C \leftarrow \mathcal{E}_{N,y}(b) : A(N, y, C) = b]. \end{aligned}$$

On the other hand, when y is a quadratic residue then *the view of A is independent of the bit b being encrypted.* This is the case because regardless of the value of b, the ciphertext C will always be a (uniformly distributed) quadratic residue. Thus, even an all-powerful A can only guess the value of b with probability $1/2$, and hence the probability that A' outputs 1 in this case is exactly $1/2$.

Since deciding quadratic residuosity is assumed to be (t, ϵ)-hard (and since A' runs in roughly time t), it must be the case that:

$$\left| \Pr[(N, y) \leftarrow \mathcal{K}; b \leftarrow \{0, 1\}; C \leftarrow \mathcal{E}_{N,y}(b) : A(N, y, C) = b] - \frac{1}{2} \right| \leq \epsilon.$$

But this is equivalent to $\mathsf{Adv}_A \leq 2\epsilon$, which is what we needed to prove. ■