Introduction to Modern Cryptography

Benny Chor

Quadratic Residues
The Discrete Logarithm Problem

Diffie & Hellman Key Exchange

Lecture 5

Tel-Aviv University

17 November, 2009

Quadratic Residues

- Definition: Let $m \geq 2$ be an integer and $0 \neq x \in Z_m$. We say that x is a quadratic residue modulo m if there exists an integer $y \in Z_m$ such that $y^2 \equiv x \pmod{m}$.
- Claim: Let $p > 2$ be a prime. Then there are exactly $(p-1)/2$ quadratic residue modulo p.
- Claim: Let $p > 2$ be a prime, and g a generator (primitive element) of the multiplicative group Z_p^*. The quadratic residues modulo p are exactly all the even powers of g, $g^0, g^2, \ldots, g^{2i}, \ldots, g^{p-3}$.

Quadratic Residues (QRs) in Z_p^*

- The quadratic residues form a multiplicative subgroup of Z_p^*.
- Since $p-1$ is even, the identity
  $x^{p-1}-1=\left(x^{(p-1)/2}-1\right)\cdot\left(x^{(p-1)/2}+1\right)$
  holds over any field, and in particular in Z_p^*.
- In Z_p^*, $x^{p-1}-1$ has $p-1$ distinct roots.
- Thus $x^{(p-1)/2}-1$ has $(p-1)/2$ roots in Z_p^*.

Quadratic Residues in Z_p^* (cont.)

Euler's Theorem: An element $x \in Z_p^*$ is a quadratic residue if and only if $x^{(p-1)/2} \equiv 1 \pmod{p}$.

Proof Sketch:

- Suppose $x \in Z_p^*$ is a quadratic residue. Then there is some $y \in Z_p^*$ such that $x = y^2$. Therefore, $x^{(p-1)/2} = (y^2)^{(p-1)/2} = y^{p-1} = 1$.
- Suppose $x^{(p-1)/2} \equiv 1 \pmod{p}$. Let $g \in Z_p^*$ be a primitive element, and $x = g^i$. Then $x^{(p-1)/2} = g^{i \cdot (p-1)/2}$. Since g has order $p-1$, $p-1$ must divide $i \cdot (p-1)/2$. This implies that i is even, so $x = g^i$ means x is a QR. ♠

Testing Quadratic Residues in Z_p^*

Given $x \in Z_p^*$, we want to determine (algorithmically) if it is a quadratic residue or not.
The inputs are x and p (specifies the field). Input length is $2 \log p$.

- An inefficient algorithm:
  - Go over all $y \in Z_p^*$, and for each of them test if $x = y^2$.
  - Complexity is $O(p)$ (plus some small change). This is exponential in the input length.

Testing Quadratic Residues in Z_p^* (cont.)

Given $x \in Z_p^*$, we want to determine (algorithmically) if it is a quadratic residue or not.
The inputs are x and p (specifies the field).

- An efficient algorithm:
  An element $x \in Z_p^*$ is a quadratic residue if and only if $x^{(p-1)/2} \equiv 1 \pmod{p}$.
  - Using repeated squaring, we compute $x^{(p-1)/2} \pmod{p}$, and check if it equals 1.
  - This takes $O(\log p)$ multiplications modulo p.
  - Each modular multiplication takes $O(\log^2 p)$ bit operations, so overall we have $O(\log^3 p)$ bit operations, which is cubic in the input length.

Quadratic Residues in Z_m^* (m non-prime)

- Definition: Let $m \geq 2$ be an integer and $0 \neq x \in Z_m$. We say that x is a quadratic residue modulo m if there exists an integer $y \in Z_m$ such that $y^2 \equiv x \pmod{m}$.
- When m is not a prime, it is not known how to efficiently test if x is a QR modulo m.
- When m is not a prime, but its prime factorization is known, it is known how to efficiently test if x is a QR modulo m.
- Of special interest is the case where $m = p \cdot q$ is the product of two primes but its factorization is unknown.

Quadratic Residues in Z_m^* ($m = p \cdot q$, p and q primes)

- In this case, x is a QR modulo m iff it is a QR modulo both p and q.
- So if the factorization of m is known, it is possible to efficiently test QR modulo m.
- However if the factorization is unknown, then determining QR modulo m is believed to be computationally hard (but not NP-hard).
- In fact, a probabilistic encryption scheme based on the presumed difficulty of this problem was proposed (Goldwasser and Micali, 1982).

The Discrete Logarithm (DL) Problem

- Let G be a cyclic group, and g a primitive element of G.
- Let $x \in G$ be an element of the group.
- The minimal non-negative integer, i, satisfying $x = g^i$ is called the discrete log of x to base g.
- Example: discrete logs in the multiplicative group Z_p^*.

Discrete Log in Z_p^* and One Way Functions

- Let $x = g^i$ in the multiplicative group Z_p^*.
- Exponentiation can be done in $O(\log^3 p)$ bit operations.
- When $p-1$ has a large prime factor, discrete log, the inverse operation, is believed to be computationally hard.

- Under the condition on $p-1$, the mapping $i \longrightarrow g^i$ is (believed to be) a one way function.
- This is a computational notion.

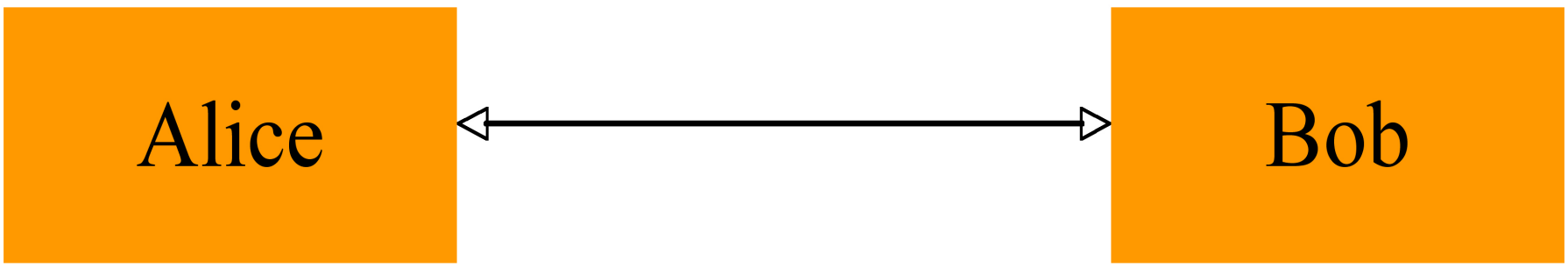
Public Key Cryptography – the New Era

"We stand today on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing . . .
. . . such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution . . .
. . . theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science."

– W. Diffie and M. Hellman, IEEE IT, vol. 22, no. 6, Nov. 1976.

Classical, Symmetric Ciphers

- Alice and Bob share the same secret key, $k_{A,B}$.
- $k_{A,B}$ must be secretly generated and exchanged prior to using the insecure channel.



Diffie and Hellman

In their seminal paper "New Directions in Cryptography", Diffie and Hellman suggest to split Bob's secret key k to two parts:

- k_E, to be used for encrypting messages to Bob.
- k_D, to be used for decrypting messages by Bob.
- k_E can be made public and be used by everybody.

This is public key cryptography, or asymmetric cryptography.

Diffie and Hellman suggested the notion of PKC, but had no concrete implementation. They did propose a concrete implementation of public key exchange.

Public Exchange of Keys

- Two parties, Alice and Bob, do not share any secret information.
- They execute a protocol, at the end of which both derive the same shared key.
- A computationally bounded eavesdropper, Eve, who overhears all communication, cannot obtain the secret key or any new information about it.
- As we did in the past, we assume Eve is passive (only listens).

Diffie and Hellman Key Exchange

- Public parameters: A large prime p (1024 bits, say) and a primitive element g in Z_p^*.
- Alice chooses at random an integer a from the interval $[0..p-2]$. She sends $x = g^a \pmod p$ to Bob (over the insecure channel).
- Bob chooses at random an integer b from the interval $[0..p-2]$. He sends $y = g^b \pmod p$ to Alice (over the insecure channel).

Diffie and Hellman Key Exchange

- Public parameters: A large prime p (1024 bits, say) and a primitive element g in Z_p^*.
- Alice chooses at random an integer a from the interval $[0..p-2]$. She sends $x = g^a \pmod p$ to Bob (over the insecure channel).
- Bob chooses at random an integer b from the interval $[0..p-2]$. He sends $y = g^b \pmod p$ to Alice (over the insecure channel).
- Alice, holding a, computes $y^a = (g^b)^a = g^{ba}$.
- Bob, holding b, computes $x^b = (g^a)^b = g^{ba}$.
- Now both have the shared secret, g^{ba}.
- (we have just witnessed a small miracle !)

Diffie and Hellman – Computational Aspects

- Public parameters: A large prime p and a primitive element g in Z_p^*.
- Should be able to efficiently find such large prime, p, and some primitive element g in Z_p^*.
- We will deal with both problems in next class, but for the time being it suffices to say that both are efficiently computable (probabilistic polynomial time computation).

Diffie and Hellman – Computational Hardness

- Computation time for exchanging the key is $O(\log^3 p)$ bit operations.

- DH key exchange is at most as secure as discrete log in Z_p^*.
- Formal equivalence between DH and DL has never been proved, though some partial results known.
- Over the last 32 years there were many attempts to crack the scheme. None succeeded, and DH key exchange (with appropriately large prime p, *e.g.* 1024 bits) is considered secure.

Key Exchange – Correctness and Security Requirements

What are the requirements from a key exchange scheme in general?

- Correctness: For each party, the combination of the public information, the communication exchanged during the execution of the protocol, and his/her private information allow the reconstruction of a key. The two participants always generate the same key.
- Secrecy: Given the public information and all the communication exchanged during the execution of the protocol, computing the shared key is computationally hard.
- Strong Secrecy: Given the public information and all the communication exchanged during the execution of the protocol, computing any efficiently computable partial information of the shared key with any non-negligible advantage is hard to compute.

Key Exchange – Correctness and Security Requirements

Does one-way relation between the private information and the communication suffice to guarantee a secure key exchange scheme in general? Not necessarily! Consider the following modification of DH:

- Public parameters: A large prime p and a primitive element g in Z_p^*.
- Alice chooses at random an integer a from the interval $[0..p-2]$. She sends $x = g^a \pmod p$ to Bob (over the insecure channel).
- Bob chooses at random an integer b from the interval $[0..p-2]$. He sends $y = g^b \pmod p$ to Alice (over the insecure channel).

Key Exchange – Correctness and Security Requirements

Does one-way relation between the private information and the communication suffice to guarantee a secure key exchange scheme in general? Not necessarily! Consider the following modification of DH:

- Public parameters: A large prime p and a primitive element g in Z_p^*.
- Alice chooses at random an integer a from the interval $[0..p-2]$. She sends $x = g^a \pmod p$ to Bob (over the insecure channel).
- Bob chooses at random an integer b from the interval $[0..p-2]$. He sends $y = g^b \pmod p$ to Alice (over the insecure channel).
- Alice, holding a, computes $g^b g^a = g^{b+a}$.
- Bob, holding b, computes $g^a g^b = g^{b+a}$.
- Now both have the shared secret, g^{b+a}.
- Finding a from g^a and b from g^b is hard.
- Does that mean Eve cannot find the key?

Do One Way Functions Hide All Partial Information?

- Consider the mapping from $[0..p-2]$ to Z_p^*, defined by $i \longrightarrow g^i \mod p$ (discrete exponentiation).
- It is believed to be one way. So it must hide much of the information about i.
- But does it hide all partial information?
- Suppose the least significant bit of i is 0. Then
- Let $g \in Z_p^*$ be a primitive element, and $x = g^i$. Then g^i is a QR mod p, and this can easily be detected using $x^{(p-1)/2} = 1$.
- Therefore, the discrete exponentiation function leaks the least significant bit of its argument.

Does DH Key Exchange Hide All Partial Information?

- From g^a and g^b, Eve could easily deduce if a and b are even or odd. The exponent arithmetic is done modulo $p-1$, which is even.
- If both a and b are odd, then $ab \pmod{p-1}$ is odd too, and g^{ba} is not a QR. If a, b, or both are even, then $ab \pmod{p-1}$ is even, so g^{ba} is a QR.
- Thus in (this original version) of DH key exchange, does leak some partial information – specifically the QR bit of the key g^{ba}.

- Possible fix (to this problem): Both a, b are picked to be odd. In addition, p is chosen to be of the form $p = 2q + 1$, where q is also a prime.

Other DH Key Exchange Systems

- The DH key exchange can be used with any underlying group, provided the discrete log in that group cannot be computed easily.
- Inappropriate group (example): The additive group $(Z_p, +)$.
- Appropriate groups (example): The multiplicative group $(Z_p, +)$, and elliptic curves.

Usage of DH Key Exchange Systems

Establishing a VPN (virtually private network):

- A DH key exchange for generating a master key.
- Master key used to encrypt session keys.
- Session key is used to encrypt traffic with a symmetric cryptosystem.
- Periodic refreshing of keys – reduced material for attacks, recovery from leaks.