

Cryptography CS 555



Lecture 15: ElGamal. DLP, CDH and DDH. Security of ElGamal

Department of Computer Sciences
Purdue University

Cristina Nita-Rotaru

Spring 2005/Lecture 15

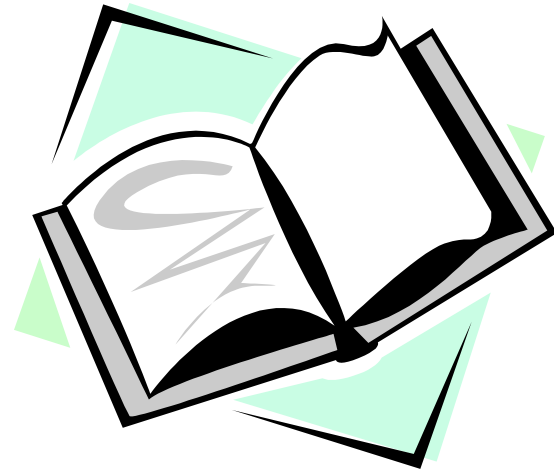
Lecture Outline

- ElGamal
- Digital signatures
 - RSA
 - ElGamal
 - DSA
 - Schnorr
- One-time digital signatures
 - Rabin
 - Lamport
 - Merkle



Recommended Reading

- Stinson: Chapter 6.1 (ElGamal)
- Stinson, Chapter 7 or HAC, Chapter 11



Discrete Logarithm

Definition

An integer g whose order modulo n is $\varphi(n)$ is called a primitive root modulo n .

Definition

Let g be a primitive root modulo n . If $\gcd(a, n) = 1$, then there exists a unique $0 < k < \varphi(n)$, such that $g^k \equiv a \pmod{n}$. k is called the index of a to base g modulo n , or the discrete logarithm of a to base g modulo n .

Discrete Logarithm Problem (DLP)

- Given a multiplicative group $(G, *)$, an element g in G having order n and an element y in the subgroup generated by g , denoted $\langle g \rangle$
- Find the unique integer x such that

$$g^x \bmod n = y$$

i.e., x is the discrete logarithm $\log_g y$

- For example, given the group Z_p^* , where p is a 1024-bit prime, let g be an element having prime order q , where q is a 160-bit prime
 - $q \mid (p-1)$
 - e.g., $Z_7^* = \{3, 2, 6, 4, 5, 1\}$, then 2 is a generator, we choose the subgroup $\{2, 4, 1\}$

Choices of Parameters

- Why use an element of order q , instead of just using a generator for Z_p^* ?
- Answer:
 - it is often beneficial to have order being a prime
 - e.g., given e , one can find d s.t. $g^{ed}=g$
 - Balance security and size
 - p needs to be large enough for discrete log to be hard, thus 1024 bits
 - we want the group to be relative small, so that an index to an element in the group is short (e.g., 160 bits)
 - it needs to be large enough to prevent exhaustive search

Algorithms for The Discrete Log Problem

- There are generic algorithms that work for every cyclic group
 - Pollard Rho
 - Pohlig-Hellman
- There are algorithms that work just for some groups such as Z_p^*
 - e.g., the index calculus algorithms
 - these algorithms are much more efficient
 - therefore, 1024 bits are needed for adequate level of security

Bit Security in Discrete Log

- Even though it is difficult to find $\log_g x$, it is possible to determine some bits in $\log_g x$
 - e.g., let g be the generator of Z_p^* , consider the least significant bit (LSB) of $\log_g x$
 - recall that $\log_g x$ is even iff. x is quadratic residue in Z_p^*
- However, finding some bits (aka. hard-core bits) is as hard as computing discrete log
 - in Z_p^* , when $p-1=2^s t$, where t is odd, computing the s least significant bits are easy, computing the $s+1$ LSB is difficult

Parameters Setup

- How to generate primes p and q s.t. $q \mid (p-1)$ and an order q element in Z_p^* ?
- Need to know the factorization of $(p-1)$
 - enables one to determine the order of any element in Z_p^* ,
- Approach 1: generate a random prime p and then factor $(p-1)$
- Approach 2: generate a random q first, then choose r s.t. $p=2rq+1$ is a prime

Diffie-Hellman Key Establishment

- A and B wishes to establish a shared secret key so that no eavesdropper can compute the key:
- A and B shares public parameters a group Z_p and a generator g
 - A randomly chooses x and send $g^x \bmod p$ to B
 - B randomly chooses y and send $g^y \bmod p$ to A
 - Both A and B can compute $g^{xy} \bmod p$
 - It is (believed to be) infeasible for an eavesdropper to compute $g^{xy} \bmod p$
 - A and B can establish a shared secret without sharing any secret to start with

CDH and DDH

- Security of the Diffie-Hellman key establishment protocol based on the CDH problem
- Computational Diffie-Hellman (CDH)
 - Given a multiplicative group $(G, *)$, an element $g \in G$ having order q , given g^x and g^y , find g^{xy}
- Decision Diffie-Hellman (DDH)
 - Given a multiplicative group $(G, *)$, an element $g \in G$ having order q , given g^x , g^y , and g^z , determine if $g^{xy} \equiv g^z \pmod n$
- Discrete Log is at least as hard as CDH, which is at least as hard as DDH.

ElGamal

- Published in 1985 by ElGamal
- Its security based on the intractability of the DLP and the CDH and DDH problem
- Message expansion: the ciphertext is twice as big as the original message
- Uses randomization, each message has $p-1$ possible different encryptions

El Gamal

Key Generation

- Generate a large random prime p such that DLP is infeasible in \mathbb{Z}_p and a generator g of the multiplicative group \mathbb{Z}_p of the integers modulo p
- Select a random integer a , $1 \leq a \leq p-2$, and compute
$$y = g^a \bmod p$$
- Public key is $(p; g; y = g^a)$
- Private key is a .

ElGamal (cont.)

Encryption:

Message M into ciphertext C

Select a random integer k , $0 < k < p-2$.

Compute $\alpha = g^k \bmod p$ and $\beta = M \alpha^k \bmod p$.

Ciphertext $C = (\alpha, \beta)$

Decryption:

Compute α^{-a} as follows: $\alpha^{p-1-a} \bmod p = \alpha^{-a} \bmod p$

$M = \alpha^{-a} \beta \bmod p$

WHY DECRYPTION WORKS?

$$\alpha^{-a} \beta \bmod p \equiv g^{-ka} M \cdot (g^a)^k \bmod p \equiv M \bmod p$$

Parameters Size

- All parties could use the same modulus p and generator g
- Different encryptions should use different k
- Prime p should be chosen as 1024 bits to ensure that DLP is infeasible, while k should be 160 bits
- Algorithm can also be defined in groups other than Z_p^*
 - e.g., in elliptic curves

Security of ElGamal

- ElGamal is not semantically secure.
- WHY? An attacker can learn information about the plaintext without decrypting: given two encryptions, can say which plaintext was a quadratic residue and which one was not.

Semantically Secure ElGamal

- Choose p such that $p = 2q + 1$, where q is also prime
- Then define ElGamal in Q_q , the subgroup of quadratic residues modulo p , this subgroup is a cyclic subgroup of Z_p having order q
- Equivalent with restricting the message m , α^a and $y_1 = \alpha^k \bmod p$ to be quadratic residues

ElGamal and DH Problems

- Semantic security of ElGamal is equivalent to the infeasibility of Decision Diffie-Hellman
- ElGamal decryption (without knowing the secret key) is equivalent to solving Computational Diffie-Hellman

CDH and ElGamal

Prove that any algorithm that solves CDH can be used to decrypt ElGamal ciphertexts

Intuition: Compute m from $(\alpha = g^k, \beta = m \alpha^k)$ is equivalent to compute α^k , one knows $\alpha = g^k$, $\gamma = g^a$, and needs to compute g^{ka} .

Formal Proof: “ \Rightarrow ” Assume that algorithm OracleCDH solves CDH

and let (α, β) be an ElGamal encryption and

let public key $(g, \gamma = g^a)$ $\alpha = g^k \pmod p$, $\beta = m (g^a)^k \pmod p$

$y = \text{OracleCDH}(g, \alpha, \gamma)$ and

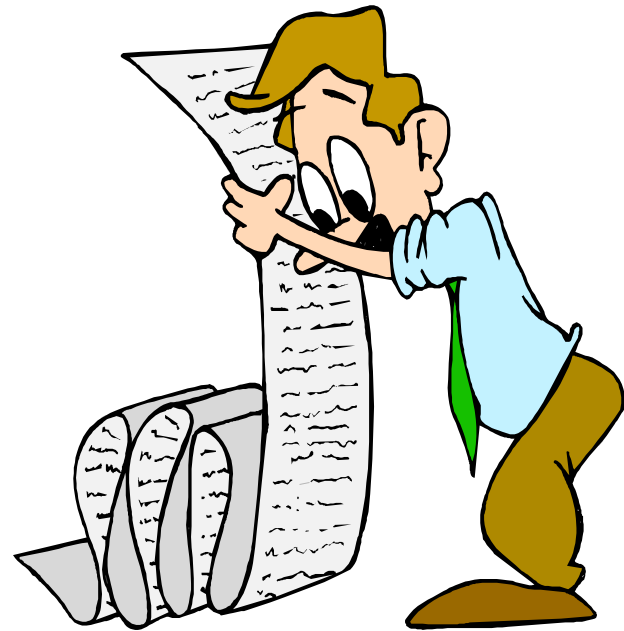
$x = \beta y^{-1}$ then x is the decryption of (α, β)

Decision D-H \Rightarrow ElGamal

- Given decision D-H oracle, find two messages whose ElGamal encryptions can be distinguished
- For any two M_0, M_1 : ($\alpha = g^a$)
 - $E(M_0) = g^x, M_0 \alpha^x, \quad E(m_1) = g^y, M_1 \alpha^y$
 - Suppose receive ciphertext (α^r)
 - Feed $\langle \alpha, \alpha^r, \alpha^r / m_0 \rangle$
 - when (α^r) is $E(M_0)$, this is $\langle g^x, g^{a+r}, M_0 g^{ax} g^{xr} / M_0 \rangle = \langle g^x, g^{a+r}, g^{x(a+r)} \rangle$
 - when (α^r) is $E(M_1)$, this is $\langle g^x, g^{a+r}, g^{x(a+r)} M_1 / M_0 \rangle$
 - if the DDH oracle say yes, we say 0, otherwise we say 1

Summary

- DDH can be reduced to CDH that can be reduced to DLP
- Semantic security of ElGamal is equivalent to solving DDH
- Decryption for ElGamal is equivalent to solving CDH



Next Lecture...

- Digital signatures
 - RSA
 - ElGamal
 - DSA
 - Schnorr

