Cristina Nita-Rotaru

CS355: Cryptography

Lecture 14: Diffie-Hellman, ElGamal

Many cryptographic algorithms rely on exponentiation
Example: Diffie-Hellman key exchange, ElGamal encryption

a^x mod n, where x is supposed to be secret

QUESTIONS:

1) how difficult is to compute x from a^x mod n
2) from a^x mod n and a^y mod n how easy it to compute a^{xy} mod n

Logarithm: $\log_a b = x$, where $a^x = b$
Discrete logarithm: x with property that $a^x \bmod n = b$

Groups

Definition

A *group* $(G, *)$ is a set G on which a binary operation is defined which satisfies the following axioms:

Closure: For all $a, b \in G$, $a * b \in G$.

Associative: For all $a, b, c \in G$, $(a * b) * c = a * (b * c)$.

Identity: $\exists$ e $\in G$ s.t. for all $a \in G$, $a *$ e $= a =$ e $* a$.

Inverse: For all $a \in G$, $\exists\, a^{-1} \in$ G s. t. $a * a^{-1} = a^{-1} * a =$e.

Example

$(Z_n, +)$ is a group, where + is addition modulo n

$(Z_{p,*})$ = is a group, where * is multiplication modulo p

Groups (cont.)

Definition:

A group $(G, *)$ is called an *abelian group* if operation $*$ is a commutative operation:

Commutative: For all $a, b \in G$, $a * b = b * a$.

Example:

(R, +) is an abelian group

Definition

A group G is *cyclic* if $\exists$ g $\in$G s.t. any h $\in$G can be writen $h = g^i$.

g is called group generator.

Example

Cyclic groups: $(Z_2, *)$, $(Z_3, *)$

Cristina Nita-Rotaru

Order of a Group

Definition

The *order* of a group G, ord(G), is defined as the number of elements in the group.

Definition

A group G is *finite,* if |G| = ord(G), is finite.

We can show that the order of $(Z_n, *)$ is $\Phi(n)$

Example:

What is the order of $(Z^*_7, *)$, $(Z^*_{700}, *)$?

Order of an Element

Definition

The *order of an element g* from a finite group G, is the smallest power of *n* such that $g^n=e$, where *e* is the identity element.

Example:

What is the order of 2 in $(Z^*_5, *)$?

It is 4 because $2^4 \equiv 1 \bmod 5$

What is the order of 3 in $(Z^*_{10}, *)$?

It is 4 because $3^4 \equiv 1 \bmod 10$

OBS: order of an element modulo n =< $\Phi(n)$

Primitive Root

Definition

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

Example

$(Z_7^*, *)$, $5^6 \equiv 1 \bmod 7$ and $\Phi(7) = 6$

$5^6 = 15625$

$(Z_8^*, *)$ does not have a primitive root

FACT

The group $G = \langle Z_n^*, * \rangle$ has primitive roots only if n is 2, 4, p^t or $2p^t$ where p is an odd prime number.

Primitive Roots and Cyclic Groups

FACT

If a group (Z_n*, *) has a primitive root, it is cyclic. Each primitive root is a generator and can be used to create the whole set. $Z_n^* = \{g_1, g^2, \ldots g^{\Phi(n)}\}$

FACT

If the group (Z_n*, *) has any primitive root, the number of primitive roots is $\Phi(\Phi(n))$

OBSERVATION

(Z_n*, *) is cyclic if it has primitive roots

(Z_p*, *) is always cyclic

Discrete Logarithm

Definition

Let $G = (Z_n^*, *)$ be a cyclic group with generator (primitive root) g. Then every element a of G can be written as $g^k \equiv a \bmod n$.

k is called the index of a base g modulo n, or the discrete logarithm of a to base g modulo n.

Discrete logarithms behave like traditional logarithms.

$\log_g 1 \equiv 0 \bmod \Phi(n)$

$\log_g xy \equiv (\log_g x + \log_g y) \bmod \Phi(n)$

$\log_g x^k \equiv k \log_g y \bmod \Phi(n)$

$(Z_p^*, *)$

1, 2, … p-1
It always has primitive roots
It is cyclic
The primitive root is the base of the discrete logarithm

Diffie-Hellman Key Establishment

- A and B wish to establish a shared secret key without sharing any secret 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 sends g^x mod p to B
  - B randomly chooses y and sends g^y mod p to A
  - Both A and B can compute g^{xy} mod p

  - It is (believed to be) infeasible for an eavesdropper to compute g^{xy} mod p
  - DLP must be difficult to compute in Z_p

Diffie-Hellman Example

$p = 11,\ g = 2$

Alice selects random x and sends Bob:

$$A = g^x \bmod p.$$

$$x = 4,\ A = 2^4 \bmod 11 = 16 \bmod 11 = 5$$

Bob generates random y and sends Alice:

$$B = g^y \bmod p.$$

$$y = 6,\ B = 2^6 \bmod 11 = 64 \bmod 11 = 9$$

Alice calculates secret key: $K = (B)^x \bmod p.$

$$K = 9^4 \bmod 11 = 6561 \bmod 11 = 5.$$

Bob calculates secret key: $K = (A)^y \bmod p.$

$$K = 5^6 \bmod 11 = 15625 \bmod 11 = 5.$$

Example from Tom Dunigan's notes: http://www.cs.utk.edu/~dunigan/cs594-cns00/class14.html

Discrete Logarithm Problem (DLP)

Given a multiplicative group (G, *), and a primitive root g in G and an element y, find the unique integer x such that

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

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

Algorithms for The Discrete Log Problem (DLP)

- 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
  - **1024 bits for p are needed for adequate level of security**

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, *), and a primitive root $g \in G$, given $g^x \bmod n$ and $g^y \bmod n$, find $g^{xy} \bmod n$
- Decision Diffie-Hellman (DDH)
  - Given a multiplicative group (G, *), and a primitive root $g \in G$, given $g^x \bmod n$, $g^y \bmod n$, and $g^z \bmod n$, determine if $g^{xy} \equiv g^z \bmod n$
- DLP is at least as hard as CDH, which is at least as hard as DDH.

ElGamal

- Published in 1985 by ElGamal
- Its security is 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 Z_p and a generator g of the multiplicative group Z_p of the integers modulo p
- Select a random integer a, $1 \leq a \leq p\text{-}2$, and compute

$$g^a \bmod p$$

- Public key is (p; g ; $\beta = g^a \bmod p$)
- Private key is a.

ElGamal (cont.)

Encryption:

Message M into ciphertext C

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

Compute $\gamma = g^k \bmod p$ and $\delta = M\,\beta^k \bmod p$.

Ciphertext $C = (\gamma, \delta)$

Decryption:

Compute γ^{-a} as follows: $\gamma^{p-1-a} \bmod p = \gamma^{-a} \bmod p$

$M = \gamma^{-a}\,\delta \bmod p$

WHY DECRYPTION WORKS?

$\gamma^{-a}\,\delta \bmod p \equiv g^{-ka} M \,.(g^a)^k \bmod p \equiv M \bmod p$

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

ElGamal Example

$g = 2$, $p=13$.

secret key $a = 7$

public key $\beta = g^a \textit{ mod } p = 2^7 \text{ mod } 13 = 11$.

Encrypt message $M = 3$.

Select a random $k = 5$ and

$\gamma = g^k \text{ mod } p = 2^5 \textit{ mod } 13 = 6$

$\delta = M \beta^k \text{ mod } p = 3 * 11^5 \text{ mod } 13 = 3 * 7 \text{ mod } 13 = 8$

Ciphertext $C = (\gamma, \delta) = (6, 8)$

Decrypt $\gamma^{p-1-a} \text{ mod } p = \gamma^{-a} \text{ mod } p = 6^{13-1-7} \text{ mod } 13 = 6^5 \text{ mod } 13 = 7776 \text{ mod } 13 = 2$

$M = 2 * 8 \text{ mod } 13 = 16 \text{ mod } 13 = 3$

Example courtesy of http://www.cs.chalmers.se/Cs/Grundutb/Kurser/krypto/lect05.pdf

Optional homework

- Encrypt m = 7, k=5
- Encrypt m = 2, k = 3

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 $y1 = \alpha^k \bmod p$ 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