Elgamal CryptoSystem

Murat Kantarcioglu

Cryptosystems Based on DL

- DL is the underlying one-way function for
  - Diffie-Hellman key exchange
  - DSA (Digital signature algorithm)
  - ElGamal encryption/digital signature algorithm
  - Elliptic curve cryptosystems
- DL is defined over finite groups

Discrete Logarithm Problem

- Let p be a prime and α and β be nonzero integers in Z_p and suppose

  $\beta \equiv \alpha^x \bmod p.$
- The problem of finding x is called the discrete logarithm problem.
- We can denote it as

  $x = \log_{\alpha} \beta$
  - Often, α is a primitive root mod p
- Reminder: Z_p is a field {0, 1, …, p-1}
- Addendum: Z^*_p is a cyclic finite group {1, …, p-1}

Example: Discrete Log

- Example:
  - Let p = 11, α = 2, and β = 9.
  - By exhaustive search,

x	0	1	2	3	4	5	**6**	7	8	9	10
α^x	1	2	4	8	5	10	**9**	7	3	6	1

- $\log_2 9 = 6$.
- $\beta \equiv \alpha x \bmod p$.

Computing Discrete Log

- When p is small, it is easy to compute discrete logarithms by exhaustive search.
- However, it is a hard problem to solve for primes p with more than 200 digit.
- One-way function.
  - It is easy to compute modular exponentiation
  - But, it is hard to compute the inverse operation of the modular exponentiation, i.e. discrete log.

The ElGamal PKC

- Based on the difficulty of discrete logarithm, was invented by Tahir ElGamal in 1985.
- Alice wants to send a message m to Bob.
- Bob chooses a large prime p and a primitive root α.
  - Assume m is an integer $0 < m < p$.
- Bob also picks a secret integer a and computes
  - $\beta \equiv \alpha^a \bmod p$.
- (p, α, β) is Bob's public key.
- (a) is his private key

The ElGamal PKC: Protocol

Alice	Bob
Chooses a secret integer k	
Computes $r \equiv \alpha^k \bmod p$	
Computes $t \equiv \beta^k \cdot m \bmod p$	
Sends (r, t) to Bob.	
	Computes $t \cdot r^{-a} \equiv m \bmod p$

This works since

$t \cdot r^{-a} \equiv \beta^k \cdot m \cdot (\alpha^k)^{-a} \equiv (\alpha^a)^k \cdot m \cdot (\alpha^k)^{-a} \equiv m \bmod p$

Analysis of ElGamal PKC

- a must be kept secret.
- k is a random integer,
  - β^k is also a random nonzero integer mod p.
  - Therefore, $t \equiv \beta^k \cdot m \bmod p$ is the message m multiplied by a random integer.
  - t is also a random integer
- Knowing r does not help either.
- If Eve knows k,
  - she can calculate $t \cdot \beta^{-k} \equiv m \bmod p$.
  - k must be secret

Analysis of ElGamal PKC

- A different random k must be used for each message m.
  - Assume Alice uses the same k for two different messages m_1 and m_2,
  - the corresponding ciphertexts are (r, t_1) and (r, t_2).
  - If Eve finds out the plaintext m_1, she can also determine m_2 as follows
  - $t_1/m_1 \equiv \beta^k \equiv t_2/m_2 \pmod{p} \Rightarrow m_2 \equiv (t_2 m_1)/t_1$

Semantic Security (IND-CPA for Public Key Encryption)

- The IND-CPA game

Challenger | Adversary

Challenger: picks a random key pair (K, K^{-1}), and picks random $b \in \{0,1\}$

Challenger → Adversary: K

Adversary: picks M_0, M_1 of equal length

Adversary → Challenger: M_0, M_1

Challenger → Adversary: $C = E_K[M_b]$

Adversary → Challenger: $b' \in \{0,1\}$

Attacker wins game if b=b'

Semantic Security of ElGamal

- Note that the generic ElGamal encryption scheme is not semantically secure.
- We can infer whether a ciphertext is quadratic residue or not.
- We can use the above fact to come up with two message where one of them is a quadratic residue and the other one is a quadratic non-residue so that attacker has high advantage in distinguishing encryptions.
- The above attack does not work if β, every plaintext is quadratic residue and p=2q+1 where q is prime.
  - It can be shown that this version is semantically secure if DL is infeasible.

CDH and DDH

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

CDH and ElGamal

- *Prove that any algorithm that solves CDH can be used to decrypt ElGamal ciphertexts*
- Proof Sketch: “=>” Assume that algorithm OracleCHD solves CDH and let (y_1, y_2) be an ElGamal encryption and let public key (p, α, β) and $y_1 = \alpha^k \bmod p$

  γ = OracleCDH(α, β, y_1) and

DDH => ElGamal

- Given DDH oracle, find two messages whose ElGamal encryptions can be distinguished
- For any two x_0, x_1: $(\beta = \alpha^a)$
  - $E(x_0) = \alpha^r, x_0\, \beta^r$, $E(x_1) = \alpha^t, x_1\, \beta^t$
  - Suppose receive ciphertext (y_1, y_2)
  - Feed $< y_1, \beta\alpha^b, y_2\, y_1^{\,b} / x_0 >$
  - when (y_1, y_2) is $E(x_0)$, this is $< \alpha^r, \alpha^{a+b}, x_0\alpha^{rb}\alpha^{\,rb} / x_0 > =$ $< \alpha^r, \alpha^{a+b}, \alpha^{r(a+b)} >$
    - when (y_1, y_2) is $E(x_1)$, this is $< \alpha^t, \alpha^{a+b}, \alpha^{t(a+b)} x_1 / x_0 >$
  - if the DDH oracle say yes, we say 0, otherwise we say 1