DTTF/NB479: Dszquphsbqiz Day 25

- HW6 due tomorrow
  - Teams T will get to pick their presentation day in the order $\underset{p \in T}{avg}\, lateDaysLeft(p)$
  - Teams mostly formed. One team of 2 or two teams of 3?

- Questions?
- Review of mid-term feedback

- This week:
  - Discrete Logs, Diffie-Hellman, ElGamal
  - Hash Functions

Discrete Logs

Given $\beta = \alpha^x \pmod{p}$

Find x

We denote this as $x = L_\alpha(\beta)$

Why is this hard?

Some things we won't cover in class about Discrete Logs

- 7.2.2 Baby step, Giant Step (worth reading)
- 7.2.3 Index Calculus: like sieve method of factoring primes
  - The equation on p. 207 might help with some of homework 7.

$$\alpha^k \equiv \prod p_i^{a_i} \pmod{p}$$
$$\Rightarrow k \equiv \sum a_i L_\alpha(p_i) \pmod{p-1}$$

- Discrete logs mod 4 and bit commitment

Diffie-Hellman is an alternative to RSA for key exchange, but is based on discrete logs

- Publish large prime p, and a primitive root α
- Alice's secret exponent: x
- Bob's secret exponent: y
  - $0 < x,y < p-1$
- Alice sends $\alpha^x \pmod{p}$ to Bob
- Bob sends $\alpha^y \pmod{p}$ to Alice
- Each know key $K=\alpha^{xy}$
- Eve sees p, α^x , α^y …

  why can't she determine α^{xy}?

Diffie-Hellman Key Exchange involves three computational problems

- Publish large prime p, primitive root α
- Alice's secret exponent: x
- Bob's secret exponent: y
  - $0 < x,y < p-1$
- Alice sends α^x (mod p) to Bob
- Bob sends α^y (mod p) to Alice
- Each know key $K=\alpha^{xy}$
- Eve sees α, p, α^x , α^y ; why can't she determine α^{xy}?

- ***Discrete logs:***
  "Given $\alpha^x = \beta$ (mod p), find x
- ***Computational Diffie-Hellman problem:***
  "Given α, p, α^x (mod p), α^y (mod p), find α^{xy} (mod p)"
- ***Decision Diffie-Hellman problem:***
  "Given α, p, α^x (mod p), α^y (mod p), and $c \neq 0$ (mod p). Verify that $c=\alpha^{xy}$ (mod p)"

What's the relationship between the three? Which is hardest?

The ElGamal Cryptosystem is an entire public-key cryptosystem like RSA, but based on discrete logs

p large so secure and > m = message

↓

Bob chooses prime p, primitive root α, integer a

Bob computes $\beta \equiv \alpha^a \pmod{p}$

Bob publishes (α, p, β) and holds *a* secret

Alice chooses secret k, computes and sends to Bob the pair (r,t) where

- $r \equiv \alpha^k \pmod{p}$
- $t \equiv \beta^k m \pmod{p}$

Bob calculates: $tr^{-a} \equiv m \pmod{p}$

Why does this decrypt?

ElGamal Cryptosystem

Bob publishes (α, p, $\beta \equiv \alpha^a$)

Alice chooses secret k, computes and sends to Bob the pair (r,t) where

- $r \equiv \alpha^k \pmod p$
- $t \equiv \beta^k m \pmod p$

Bob finds: $tr^{-a} \equiv m \pmod p$

- Why does this work?

- Multiplying m by β^k scrambles it.
- Eve sees α, p, β, r, t. If she only knew a or k!
  - Knowing a allows decryption.
  - Knowing k also allows decryption. Why?
- Can't find k from r or t. Why?

ElGamal

Bob publishes (α, p, $\beta \equiv \alpha^a$)

Alice chooses secret k, computes and sends to Bob the pair (r,t) where

- $r \equiv \alpha^k \pmod p$
- $t \equiv \beta^k m \pmod p$

Bob finds: $tr^{-a} \equiv m \pmod p$

1. Show that Bob's decryption works √
2. Eve would like to know k. Show that knowing k allows decryption. Why? √
3. Why can't Eve compute k from r or t? √
4. Challenge: Alice should randomize k each time. If not, and Eve gets hold of a plaintext / ciphertext (m_1, r_1, t_1), she can decrypt other ciphertexts (m_2, r_2, t_2). Show how.
5. If Eve says she found m from (r,t), can we verify that she really found it, using only m,r,t, and the public key (and not k or a)? Explain.
6. (For HW: Create a public key (α, p, β), encrypt a message as (r,t), and decrypt it using the private key. You may do this with a friend as we did for RSA, or do it on your own.)