Diffie-Hellman Key Exchange

CS355 September 29, 2016

Diffie-Hellman Key Exchange

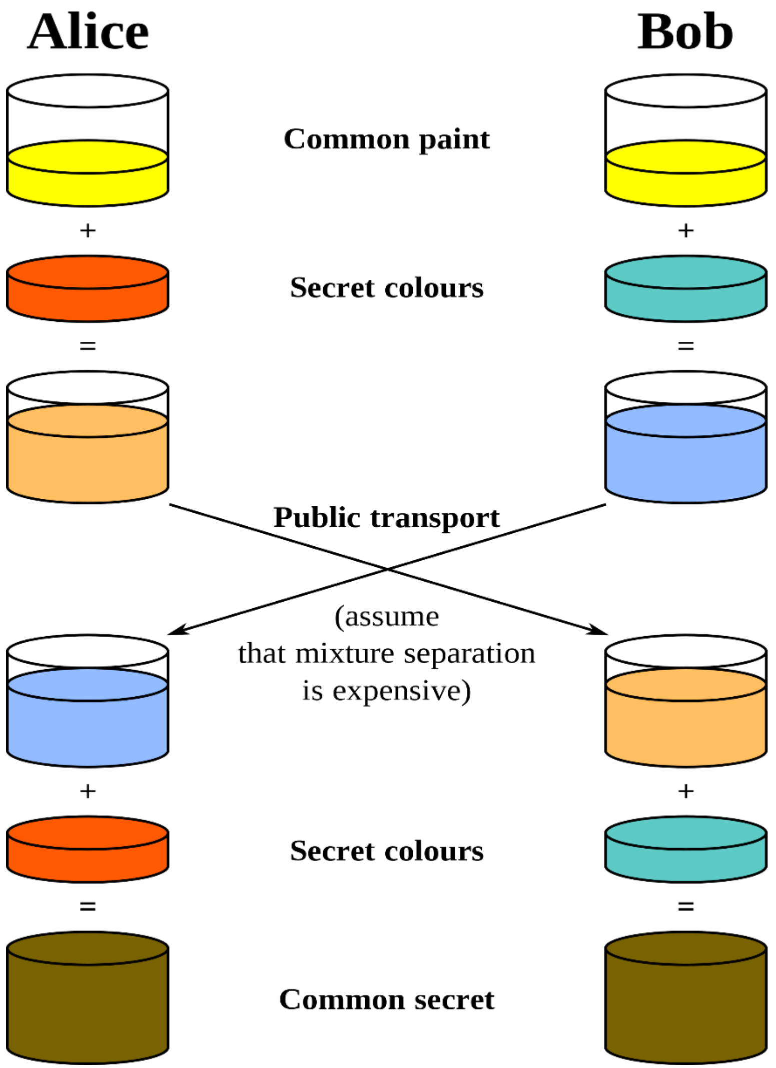
- Allows two parties to choose a common secret key, say for use in DES or AES, over an insecure channel (assumes eavesdroppers).

- Based on the hardness of the Discrete Logarithm Problem.

- In what applications is D-H Key Exchange most commonly used?

- Well, if the process is secure over an insecure channel, why don’t we just use it for encrypting messages, rather than just key material?

Diffie-Hellman Key Exchange Process



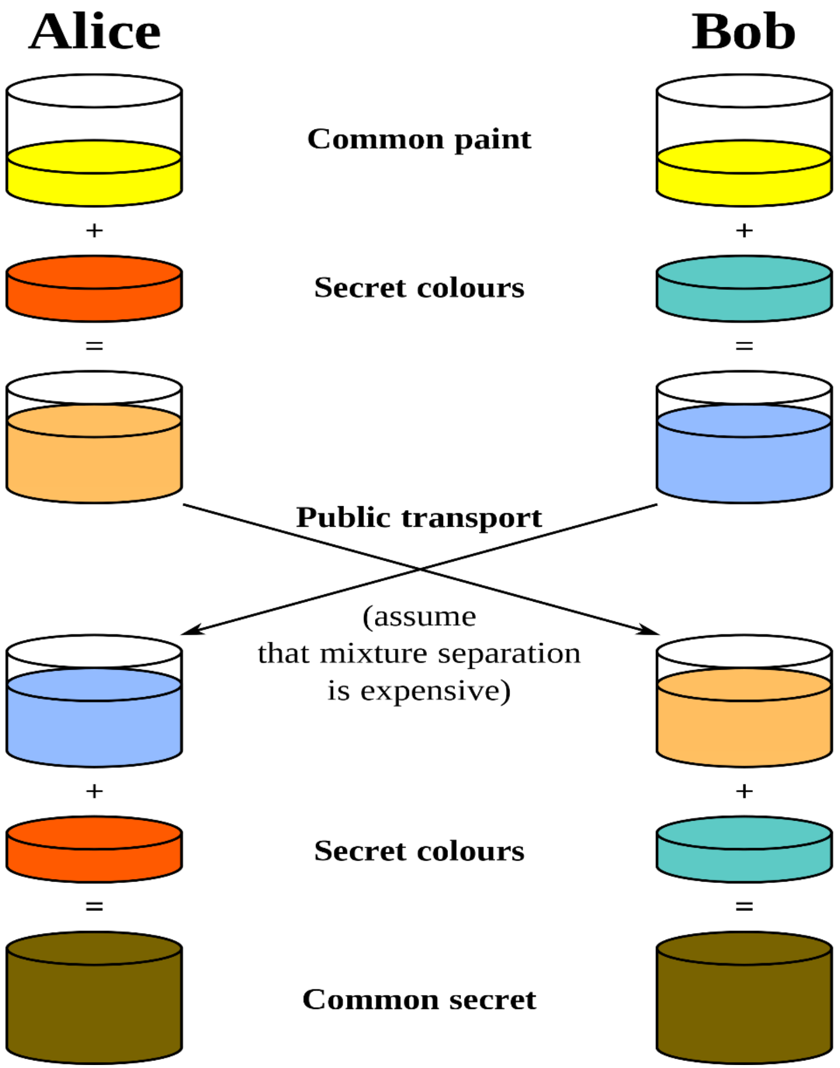
1. Two users agree on a common large prime, p, and a constant value, a. These values are public. They are the "Common Paint"

2. Alice secretly chooses a random x_a in $0 < x_a < p - 1$ and computes $y_a = a^{x_a} \ (mod\ p)$. Bob secretly chooses a random x_b in $0 < x_b < p - 1$ and computes $y_b = a^{x_b} \ (mod\ p)$.

3. Alice sends y_a to Bob. Bob sends y_b to Alice. An eavesdropper, knowing p and a, and seeing y_a and y_b, can only compute x_a or x_b if she can solve the Discrete Log problem quickly.

https://upload.wikimedia.org/wikipedia/commons/thumb/4/46/Diffie-Hellman_Key_Exchange.svg/2000px-Diffie-Hellman_Key_Exchange.svg.png

Diffie-Hellman Key Exchange Process



4. Alice uses her secret key and the information received from Bob to compute: $K_A = y_B^{x_A} \; (mod\ p)$.
   - Bob uses his secret key and the information received from Alice to compute: $K_B = y_A^{x_B} \; (mod\ p)$.
   - Then…

$$K_A \equiv a^{x_a \times x_b} \equiv K_B$$

   - And…

$$0 < K_A, K_B < p - 1$$

   - So…

$$K_A = K_B$$

https://upload.wikimedia.org/wikipedia/commons/thumb/4/46/Diffie-Hellman_Key_Exchange.svg/2000px-Diffie-Hellman_Key_Exchange.svg.png

Diffie-Hellman Key Exchange: Summary

- Now Alice and Bob have an agreed-upon symmetric key that they can use in DES or AES, but….

- D-H Key Exchange would be broken if we could compute discrete logs quickly.

Diffie-Hellman Key Exchange Group Activity (15 minutes)

Rules:

- No computers
- Defend your answer with the math in the previous slides
- Prepare to explain the logic of behind your answer after 15 minutes

- The newest version of Internet Explorer is available for Beta testing. Alice begins to use it, but realizes that, when seeking to establish a TLS connection, it sends Alice's secret key.
  - Whose information is now vulnerable in this TLS connection (Alice's, Bob's, both)?
  - Why?

Discrete Logarithms

CS355 September 29, 2016

Discrete Logarithms

- Solving Discrete Logarithms quickly is the obvious attack on Diffie Hellman.
- The Discrete Logarithm Problem
  - Solving $a^x \equiv b \ (mod\ p)$ for x is hard.

Remember Regular Logarithms?

- A logarithm is how many times you have to multiply a number by itself to get a number, b.
- The concept is the same when used in cryptography, except logarithms are created modulus some number, rather than simply in base 10.
- For example…

$$10^3 \equiv 30 \ (mod\ 97)$$

- So, the discrete log is 3
- Discrete logs have many of the same properties as regular logs…

Discrete Logarithms

- By analogy to ordinary logarithms, we may write $x = log_{a^b}$ when p (the modulus) is understood from the context.

- These discrete logarithms enjoy many properties of ordinary logarithms, such as $log_{a^{bc}} = log_{a^b} + log_{a^c}$, except that the arithmetic with logarithms must be done modulo p − 1 because $a^{p-1} \equiv 1\ (mod\ p)$.

Discrete Logarithms (Solving)

- Neglecting powers of log p, the congruence may be solved in O(p) time and O(1) space by raising a to successive powers modulo p and comparing each with b.

- So, then, the problem may also be solved in O(1) time and O(p) space by looking up b in a precomputed table of pairs (b, a^b mod p) sorted by the second coordinate.

Discrete Logarithms: Individual Activity (10 minutes)

Rules:

- No computers
- Prepare to put answers on the board after 10 minutes
- Time your answers to Q1 in order to answer Q2

1. Solve the following Discrete Logs
    - $2^x \equiv 6 \ (mod\ 13)$
    - $2^x \equiv 15 \ (mod\ 19)$
    - $5^x \equiv 20 \ (mod\ 103)$

2. Provide a written answer for these questions:
    - Would you (or your computer) use the same procedure to solve discrete logs if the numbers involved were 100 digits?
    - Make a guess the general relationship between the length of the numbers and the time it takes to solve discrete log problems.

Pohlig- Hellman Cipher

- A single-key cipher like DES or AES (not a public key cipher), except its mathematical basis is the Discrete Logarithm Problem.
  - Pohlig-Hellman is an “exponentiation cipher”. In this way, it’s like RSA.

- It’s slower than AES, so it’s not used as a plain single key cipher, but it’s used in other ways because of its mathematical properties.

Pohlig-Hellman Cipher: Why do we care?

- The Pohlig-Hellman cipher has the commutative property.

  1. The modulus can be public.
  2. You can calculate the decryption key from the encryption key using Extended Euclidean Algorithm with the encryption exponent and p-1 as inputs.

- The commutative property will be important to mental poker and electronic voting schemes we will study later in the semester.

Pohlig-Hellman Cipher: Implementation

- The P-H Cipher can be implemented by two different methods.
- In each case
  - Let $n = p = prime.$
  - $\varphi(p) = p - 1$ and $ed \equiv 1$(mod p-1)

- Method 1:
  - Keep $p, e, and\ d$ secret.
  - These three parameters form the key.
  - In this case, there is a single user, or one pair of users.

Pohlig-Hellman Cipher: Implementation

- Method 2:
  - p becomes public. $e, and\ d$ are secret.
  - $e, and\ d$ form the key.
  - Each user has a secret pair to safeguard her personal secrets.
  - Each pair of users who wish to communicate choose a key pair.
    - This is also the method with the important properties discussed earlier, and mathematically in the next slide.

Pohlig-Hellman Cipher: Important Properties

- Let p be a large prime and suppose users A and B have encryption algorithms E_A and E_A, and decryption algorithms D_A and D_B.
  - $E_A(M) = M^{e_A}\ (mod\ p)\ ;\ D_A(C) = C^{d_A}\ (mod\ p)\ where\ e_A d_A \equiv 1\ (mod\ p-1), etc.$

- Since the encryption and decryption algorithms are all exponentiation modulo a fixed modulus, they all *commute*. That is, they may be performed in any order and give the same result

- For example, $E_A(D_B(x)) = D_B(E_A(x))\ for\ every\ x$ because both are just $x^{e_A d_B} \equiv x^{d_B e_A}\ (\mathrm{mod\ p})$.