

# Lab 6: Diffie-Hellman Key Exchange

50.042 Foundations of Cybersecurity

Hand-out: October 24th

Hand-in: 11:59pm, November 7th

## 1 Objective

By the end of this lab, you should be able to:

- Implement Square-Multiply for large integer exponentiation
- Implement Diffie-Hellman Key Exchange
- Implement Shanks' Baby-step Giant-step method

## 2 Part I: Implementing Square and Multiply

- Implement square and multiply algorithm to exponentiate large integers efficiently in a computer. To do exponentiation of  $y = a^x \bmod n$ , we can use the following algorithm.

```
square_multiply(a, x, n)
{
    y = 1
    for( i = n_b-1 downto 0 ) //n_b is the number of bits in x
    {
        y = y^2 mod n
        if (x_i == 1) y = a*y mod n // multiply only if the bit of x at i is 1
    }
    return y
}
```

## 3 Part II: Diffie-Hellman Key Exchange (DHKE)

DHKE can be used to exchange keys between two parties through insecure channel. For this exercise, you need to find a partner where you can exchange your keys. Create a Python script that enables you to do the following protocol:

- Set-up. You and your partner need to agree on the following:
  - Choose a large prime  $p$ .
  - \* Go to <http://www.wolframalpha.com>



- \* Type in "prime closest to  $2^{80}$ ". This will give the prime closest to a number  $2^{80}$ . You can change the number to any other large integer number.
- Choose an integer  $\alpha \in 2, 3, \dots, p-2$ , which is a primitive element or generator in the group.
- Publish  $p$  and  $\alpha$ .
- Each of you and your partner needs to do the following separately:
  - You choose a private key  $a = k_{pr,A} \in 2, \dots, p-2$ , while your partner choose his own private key  $b = k_{pr,B} \in 2, \dots, p-2$
  - You compute your public key,  $A = k_{pub,A} \equiv \alpha^a \pmod{p}$ , while your partner compute his own public key  $B = k_{pub,B} \equiv \alpha^b \pmod{p}$
- Exchange your public keys.
- Compute the shared keys,  $k_{AB} = k_{pub,B}^a \equiv B^a \pmod{p}$ , or  $k_{AB} = k_{pub,A}^b \equiv A^b \pmod{p}$

## 4 Hand-in:

- Demo a key exchange using DHKE protocol.
- Use PRESENT with ECB mode from the previous lab to encrypt a message using the shared keys. Note that PRESENT requires the key to be either 80 or 128 bits.
- What's the advantage and disadvantage of DHKE?

## 5 Part III: Baby-Step Giant-Step Method

- This section introduce one method for solving discrete logarithm problem from which DHKE is based upon. An attacker can obtain the key if he can solve the discrete logarithm problem  $x = \log_{\alpha} \beta$ .

- To do this, the problem is written in a two-digit representation:

$$x = x_g m - x_b,$$

where  $0 \leq x_g, x_b < m$ . In this way, we can write the exponentiation as

$$\beta = \alpha^x = \alpha^{x_g m - x_b}$$

Rearranging the terms, we get

$$\beta \alpha^{x_b} = \alpha^{x_g m}$$



- Create a Python script to do the following:
  - Calculate  $m = \lfloor \sqrt{|G|} \rfloor$ , i.e. the size of the square root of the group order.
  - Baby-step phase: Compute and store into a file the values of  $\alpha^{x_b} \beta$ , where  $0 \leq x_b < m$ .
  - Giant-step phase: Compute and store the values of  $\alpha^{m \times x_g}$ , where  $0 \leq x_g < m$ .
  - Check the two list and see if there is a match such that:  $\alpha^{x_b} \beta = \alpha^{m \times x_g}$
  - If a match is found, calculate  $x = x_g m - x_b$

## 6 Hand-in:

- Generate 16-bit shared keys using the DHKE protocol. Try to calculate the shared key based on the known  $p$ ,  $\alpha$ , and public keys.
- Increase the number of bits to break slowly. To avoid attack using Baby-Step Giant-Steps method, how many bits should the key be in DHKE protocol?