Diffie-Hellman Key Exchange

Joe Divinagracia CS363

The Diffie-Hellman key exchange system has been one of the most influential in the world of digital security. Diffie-Hellman allows two parties to exchange information effectively and securely. It works in the following way: both parties first exchange a public key over an insecure network. It is public, so it does not need to be encrypted. They each generate their own private key, both of which are used to decrypt the message. The sender uses the message and their own private key, along with arbitrary values p and q, in the equation A = q^b mod(p), where A is the decryption/encryption key, p and q are arbitrary values, and b is their private key, and then using bitwise XOR on the key and message. This encrypts the message which can only be decrypted by the intended recipient who decrypts it by using the equation B = q^a mod(p) to get the same key that was generated by the sender and uses bitwise XOR to decrypt the message.

This topic is interesting to me because I wish to go into digital and cybersecurity. Having knowledge of how Diffie-Hellman works will help me understand other (and potentially create) more advanced and secure key exchange systems. One of the main issues with Diffie-Hellman is that the parties initially exchange public keys via an insecure network. This insecurity can make it easy for attackers to see and obtain those keys. This usually is not a big problem since the parties use large integer values for the equation. However, while it may be difficult to crack, it is not impossible. If an attacker can somehow crack the keys, they can then obtain all info exchanged over that connection and could potentially gain further access to a system.

Since this system is so refurbished and powerful, there have been no recent major developments to this technology. Many organizations utilize this technology to keep their info secret and secure when exchanging them. Due to this, it had to be incredibly secure, which it is, hence no need to really improve upon this technology too much more.

I am not planning to work directly with Diffie-Hellman in the future but am not ruling it out. Working in the security sector, it is bound to happen that I will have to interact with and perhaps implement it in some form or another.

Demo Code:

This is a rough pseudocode outline of how Diffie-Hellman works.

Main(){

BigInteger p = new BigInteger(“123456789”);//p and q are arbitrarily large ints

BigInteger q = new BigInteger(“987654321”);

BigInteger A = new BigInteger(“644541215”);//the other party’s private key

BigInteger b = new BigInteger(“256”);//used in the equation for the public key

BigInteger B = q.modPow(b,p);//my public key

BigInteger key = A.modPow(b,p);//decryption key

File file = new File(“file.ext”);

byte[] buffer = null;

try{

data = new byte[(int) file.length()];

in.read(buffer);

}

byte[] newBuffer = decrypt(data, key.toString());

//decrypt is a predefined function

}