Diffie-Hellman Key Exchange Algorithm

Introduction

Zero-knowledge proof is probabilistic proof because there is some small probability (soundness error) that allows a cheating prover to convince the verifier of a false statement. Standard techniques used to decrease the soundness error to any arbitrarily small value, but with additional computation cost. The proposed protocol is a deterministic algorithm with bounded values (not probabilistic), hence has no soundness error and no additional computation cost. The proposed protocol fulfills the ZKP properties and **protects against discrete logarithm attack and man-in-the middle attack**. The proposed algorithm serves as a key exchange algorithm with the addition to authentication services

- The discrete logarithm attack
  An interceptor (Eve) can intercept u and v. Find a from ($u = g^a \bmod p$), Find b from ($v = g^b \bmod p$); Then she can calculate ($K = g^{ab} \bmod p$). The secret key is not secret anymore. To make Diffie-Hellman safe from the discrete logarithm attack
  - **Brute Force Attack**: This involves trying all possible values of (a) or (b) until the correct one is found. This method is impractical for large values of (p) due to the exponential time complexity
- **man-in-the middle attack**
  A **man-in-the-middle (MITM) attack** is a type of cyberattack where an attacker secretly intercepts and possibly alters the communication between two parties who believe they are directly communicating with each other.
  - **Wi-Fi Eavesdropping**: Attackers can set up rogue Wi-Fi hotspots or exploit unsecured public Wi-Fi to intercept data

Why use Diffie-Hellman Key Exchange Algorithm:

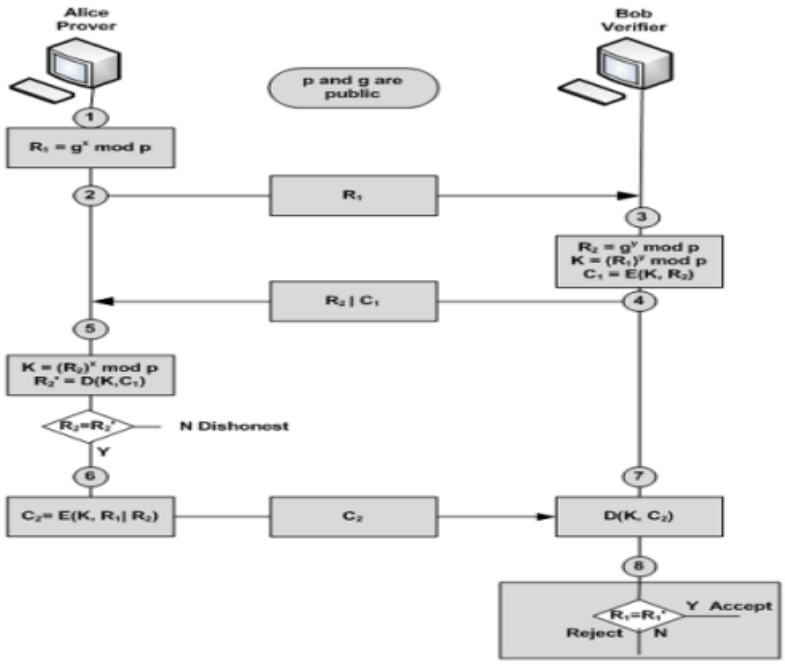
1. Secure Key Exchange: It enables secure key exchange without the need for a pre-shared secret, making it ideal for initial key establishment.
2. Scalability: It can be used in various scenarios, from small-scale personal communications to large-scale enterprise systems.
3. Foundation for Other Protocols: It serves as the basis for many other cryptographic protocols and systems, enhancing overall security.
4. Resistance to Eavesdropping: Even if an attacker intercepts the communication, they cannot derive the shared secret without solving the discrete logarithm problem, which is computationally infeasible.

Real-Life Applications:

1. Internet Security Protocols: Widely used in protocols like SSL/TLS to secure web communications, ensuring that data transmitted between a user's browser and a web server remains private
2. Virtual Private Networks (VPNs): Used in VPNs to establish secure connections over the internet, protecting data from interception and tampering
3. Secure Messaging Apps: Employed in secure messaging applications to encrypt messages, ensuring that only the intended recipient can read them
4. Wireless Security: Utilized in securing wireless communications, such as in WPA3 for Wi-Fi networks, to protect against unauthorized access

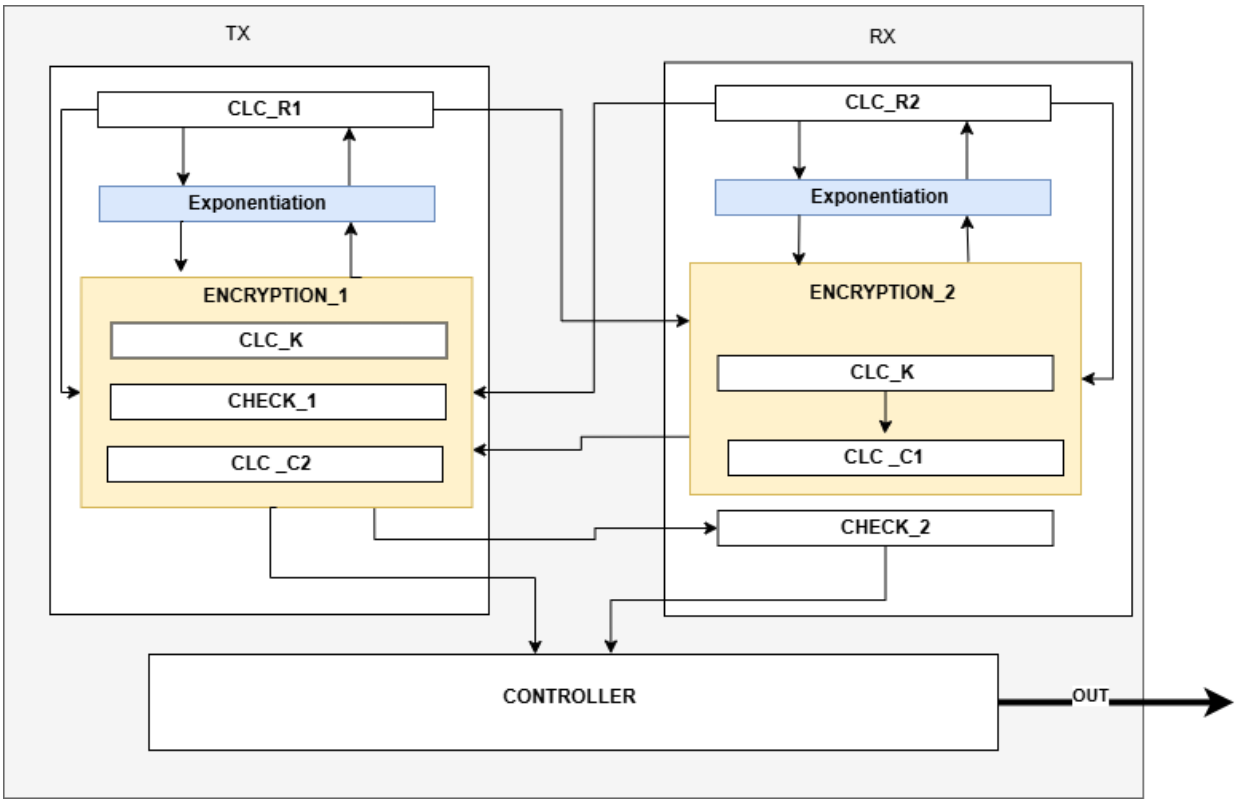
How it Works

The prover (Alice) proves to the verifier (Bob) that she knows a secret by calculating the key (K) and resend Bob's reply (R2) to the verifier (Bob) encrypted with the generated secret key (K). Bob will encrypt his own reply (R2) with the generated secret key (K) and match the two encrypted information, if matched then Alice is verified, otherwise it is rejected. The verifier also needs to prove to the prover that he is honest by sending his reply R1togather with encrypted R1, then the verifier decrypt R1' by his key and match R1and R1, if they matched then the verifier is honest



Design and Implementation:

Arch



Specification:

1. Any start flag gets high remain high till the end of operations
2. At the rest r1 and r2 take different values
3. At the rest c1 and c2 take different values
4. The prime number p must be very large
5. The numbers x and y must be large random numbers