1. \*\*Data Encryption Standard (DES):\*\*

- DES is a symmetric-key algorithm used for encryption and decryption of electronic data.

- It uses a 56-bit key and operates on blocks of data, typically 64 bits in size.

- Developed in the 1970s, it was widely used for secure communication until its vulnerabilities to brute-force attacks became apparent.

- Despite its vulnerabilities, DES laid the groundwork for modern encryption techniques.

2. \*\*Rivest-Shamir-Adleman (RSA):\*\*

- RSA is an asymmetric cryptographic algorithm used for secure data transmission.

- It's based on the difficulty of factoring large integers, which forms the basis of RSA's security.

- RSA involves a public key for encryption and a private key for decryption.

- It's widely used in secure communications, including HTTPS, SSH, and other protocols.

- RSA's security relies on the computational difficulty of factoring the product of two large prime numbers.

The Diffie-Hellman key exchange algorithm is a method used to securely establish a shared secret between two parties over an insecure channel. Here’s a simplified explanation with an example:

1. \*\*Setup\*\*:

- Alice and Bob agree on two things beforehand: a prime number \(p\) and a base \(g\), where \(g\) is a primitive root modulo \(p\).

2. \*\*Key Generation\*\*:

- Alice chooses a secret integer \(a\) and calculates \(A = g^a \mod p\).

- Bob chooses a secret integer \(b\) and calculates \(B = g^b \mod p\).

3. \*\*Exchange\*\*:

- Alice sends \(A\) to Bob.

- Bob sends \(B\) to Alice.

4. \*\*Key Calculation\*\*:

- Alice computes the shared secret key \(s\) using \(B^a \mod p\).

- Bob computes the shared secret key \(s\) using \(A^b \mod p\).

5. \*\*Result\*\*:

- Both Alice and Bob now have the same shared secret key \(s\), which they can use for secure communication, and an eavesdropper would find it computationally infeasible to determine this key.

For example, let’s say:

- \(p = 23\) (a prime number)

- \(g = 5\) (a primitive root modulo \(p\))

Alice chooses \(a = 6\) and calculates \(A = 5^6 \mod 23 = 8\). She sends \(A = 8\) to Bob.

Bob chooses \(b = 15\) and calculates \(B = 5^{15} \mod 23 = 19\). He sends \(B = 19\) to Alice.

Now, Alice calculates the shared secret key \(s\) using \(B^a \mod p\):

\(s = 19^6 \mod 23 = 2\).

Similarly, Bob calculates the shared secret key \(s\) using \(A^b \mod p\):

\(s = 8^{15} \mod 23 = 2\).

Now, both Alice and Bob have the same shared secret key \(s = 2\), which they can use for secure communication.

The Digital Signature Algorithm (DSA) is a cryptographic algorithm used for generating and verifying digital signatures. It involves creating a pair of keys: a private key for signing and a public key for verification.

Here's a simplified explanation:

1. \*\*Key Generation\*\*:

- Generate a random private key \( (x) \) and calculate the corresponding public key \( (y = g^x \mod p) \), where \( g \) is a generator, \( p \) is a large prime number, and \( \mod \) denotes the modulo operation.

2. \*\*Signing\*\*:

- Choose a random number \( k \) and calculate \( r = (g^k \mod p) \mod q \), where \( q \) is a prime divisor of \( p – 1 \).

- Calculate \( s = k^{-1}(H(m) + xr) \mod q \), where \( H(m) \) is the hash of the message \( m \).

- The signature is the pair \( (r, s) \).

3. \*\*Verification\*\*:

- Calculate \( w = s^{-1} \mod q \).

- Calculate \( u\_1 = (H(m)w) \mod q \) and \( u\_2 = (rw) \mod q \).

- Calculate \( v = ((g^{u\_1}y^{u\_2} \mod p) \mod q) \).

- If \( v = r \), the signature is valid.

Example:

Suppose we have:

- \( p = 23 \), \( q = 11 \), \( g = 2 \) (generator)

- Private key \( x = 6 \) (randomly chosen)

- Public key \( y = g^x \mod p = 2^6 \mod 23 = 64 \mod 23 = 4 \)

- Message \( m = \text{“Hello”} \)

- Hash of \( m \), \( H(m) = \text{SHA-256}(\text{“Hello”}) \)

Signing:

- Choose a random \( k = 3 \)

- Calculate \( r = (2^3 \mod 23) \mod 11 = 8 \mod 11 = 8 \)

- Calculate \( s = 3^{-1}(H(m) + xr) \mod 11 \)

- Verify the signature by performing the steps in the verification process.

This is a simplified explanation; in practice, DSA involves more steps and considerations for security.

D.

1. \*\*Time-based OTP (TOTP)\*\*: TOTP generates a one-time password based on the current time and a shared secret key. It’s typically used in two-factor authentication (2FA) systems. An example is Google Authenticator, where a secret key is shared between the server and the user’s device. The device uses this key and the current time to generate a unique password that changes every few seconds.
2. \*\*HMAC-based OTP (HOTP)\*\*: HOTP generates a one-time password using HMAC (Hash-based Message Authentication Code) with a shared secret key and a counter value. The counter value is typically incremented with each login attempt. An example is RSA SecurID, where the device and the server share a secret key, and the device generates a password based on this key and a counter value.