Q1. Data Encryption Standard (DES) and Rivest-Shamir-Adleman (RSA) Algorithms

Data Encryption Standard (DES):

Description: DES is a symmetric-key algorithm used for encrypting digital data.

Key Length: DES uses a 56-bit key, which is considered insecure for modern applications.

Influence: Despite its short key length, DES significantly influenced the field of cryptography.

History: Developed by IBM in the early 1970s, it became an official Federal Information Processing Standard (FIPS) in 1977.

Security Concerns: Vulnerable to brute-force attacks due to its short key length.

Legacy: Triple DES (3DES) is a variant that provides better security by applying DES three times.

Rivest-Shamir-Adleman (RSA) Algorithm:

Asymmetric Cryptography: RSA is an asymmetric algorithm, meaning it uses two keys: a public key for encryption and a private key for decryption.

Key Generation: RSA keys consist of a public key (e, n) and a private key (d, n).

Security Basis: RSA relies on the difficulty of factoring large integers.

Usage: Used for digital signatures, key exchange, and secure communication.

Key Sizes: RSA keys are typically 1024 or 2048 bits long.

Authentication: RSA ensures authenticity and integrity through digital signatures.

Q2. Diffie-Hellman Key Exchange Algorithm

Overview:

Diffie-Hellman (DH) allows two parties to establish a shared secret over an insecure channel.

It enables secure key exchange without prior knowledge of each other’s secrets.

How It Works:

Alice and Bob agree on a prime number (P) and a primitive root (G).

Both choose private keys (a for Alice, b for Bob).

They compute public values:

Alice: ($x = G^a \mod P$)

Bob: ($y = G^b \mod P$)

They exchange public keys (x and y).

The shared secret key is computed:

Alice: ($k_a = y^a \mod P$)

Bob: ($k_b = x^b \mod P$)

Both have the same secret key (($k_a = k_b$)).

Example:

P = 23, G = 9

Alice: ($x = 9^4 \mod 23 = 6$)

Bob: ($y = 9^3 \mod 23 = 16$)

Shared secret: 9

Q3. Digital Signature Algorithm (DSA)

Overview:

DSA provides digital signatures for authentication and data integrity.

Based on the difficulty of the discrete logarithm problem.

How It Works:

Hash the message to create a hash code.

Encrypt the hash code using the sender’s private key to create the signature.

Send both the message and the signature to the recipient.

The recipient verifies the signature using the sender’s public key.

Q4. One-Time Password (OTP) Algorithms

a. Time-based OTP (TOTP):

Generates unique passcodes every 30-60 seconds.

Synchronized between the server and the user’s device.

Used for multi-factor authentication (MFA).

b. HMAC-based OTP (HOTP):

Generates OTPs based on a counter.

Requires a shared secret key.

Used for OTP tokens and hardware tokens.