

عنوان الدرس: اليوم: التاريخ: ١٤ / /

Subject: Day: Date: / / 20 م٢٠ / /

Q1. What digital signature and why is it important in information security?
Digital signature verifies the sender's identity and ensures the message was not altered.
It provides authenticity and integrity.

Q2. Explain the role of hashing in digital signature?
Hashing produces a fixed unique digest of the message and the signature is applied to the digest to make the process faster and secure.

Q3. What is the difference between encryption and digital signature?
Encryption hides content for confidentiality, while digital signature verifies the sender and ensures message integrity.

Q4. Describe how ECDSA ensures non-repudiation.
ECDSA provides non-repudiation because only the owner of the private key can produce the signature, so the sender can't deny it.

عنوان الدرس: اليوم: التاريخ: ١٤ / /

Subject: Day: Date: / / 20

Q5. What would happen if the message is changed after signing?

If the message changes, the hash changes and the signature verification fails.

Q6 Why is SHA-256 commonly used in blockchain system
SHA-256 is secure, collision-resistant and efficient, making it ideal for blockchain system.

Q7 Mention one real-world application of digital signature
Digital signature are used in blockchain system to sign transaction (such as cryptocurrency transfers) to ensure that the sender is the genuine owner of the wallet

Q8. Write your observation and outputs from your colab experiment

The experiment showed the message being hashed, signed with the private key and successfully verified using the public key

```
▶ from ecdsa import SigningKey, NIST256p
import hashlib

# Step 1: Define the transaction message
transaction_message = "Alhasan pays 100 coins to Salim"
print("Transaction Message:", transaction_message)

# Step 2: Generate ECDSA private and public keys
private_key = SigningKey.generate(curve=NIST256p)
public_key = private_key.verifying_key

# Step 3: Hash the transaction message
message_hash = hashlib.sha256(transaction_message.encode()).digest()
print("SHA-256 Hash:", message_hash.hex())

# Step 4: Sign the hash with the private key
signature = private_key.sign(message_hash)
print("Signature:", signature.hex())

# Step 5: Verify the signature with the public key
is_verified = public_key.verify(signature, message_hash)
print("Signature Verified:", is_verified)
```

... Transaction Message: Alhasan pays 100 coins to Salim
SHA-256 Hash: d0167dd22a1c7e8c5e0de923c82e12860c1e00a59042c7ed35d6b86f278f63b7
Signature: 41186bf3b3318fd3ee52e158adade66aeab61bea827cb59bd1b1c79c1091f61385e105600d01a49a77ad7f0c7d885ed77a3277545aa7a02d895bde78d4b65ba8
Signature Verified: True