EXPERIMENT 10

Aim -

Understanding the concepts of cryptography and guidelines for using encryption.

Theory:

1. What is cryptography?

   Cryptography is the technique of securing information and communications through use of codes so that only those persons for whom the information is intended can understand it and process it. Thus preventing unauthorized access to information. The prefix “crypt” means “hidden” and suffix “graphy” means “writing”.

   Features Of Cryptography are as follows:

   - Confidentiality: Information can only be accessed by the person for whom it is intended and no other person except him can access it.
   - Integrity: Information cannot be modified in storage or transition between sender and intended receiver without any addition to information being detected.
   - Non-repudiation: The creator/sender of information cannot deny his intention to send information at later stage.
   - Authentication: The identities of sender and receiver are confirmed. As well as destination/origin of information is confirmed.

   Types Of Cryptography: In general there are three types Of cryptography:

   - Symmetric Key Cryptography
   - Hash Functions
   - Asymmetric Key Cryptography

2. What are the Cryptographic Best practices according to OWASP?

   OWASP provides a **secure coding practices** checklist that includes 14 areas to consider in your software development life cycle. Of those secure coding practices, we’re going to focus on the top eight secure programming best practices to help you protect against vulnerabilities.

   1. **Security by Design**: Security needs to be a priority as you develop code, not an afterthought. Organizations may have competing priorities where software engineering and coding are concerned.

   2. **Password Management**: Passwords are a weak point in many software systems, which is why multi-factor authentication has become so widespread. Nevertheless, passwords are the most common security credential, and following secure coding practices limits risk.

3. **Access Control**: Take a “default deny” approach to sensitive data. Limit privileges and restrict access to secure data to only users who need it. Deny access to any user that cannot demonstrate authorization. Ensure that requests for sensitive information are checked to verify that the user is authorized to access it.

4. **Error Handling and Logging**: Software errors are often indicative of bugs, many of which cause vulnerabilities. Error handling and logging are two of the most useful techniques for minimizing their impact. Error handling attempts to catch errors in the code before they result in a catastrophic failure.

5. **System Configuration**: Clear your system of any unnecessary components and ensure all working software is updated with current versions and patches. If you work in multiple environments, make sure you’re managing your development and production environments securely.

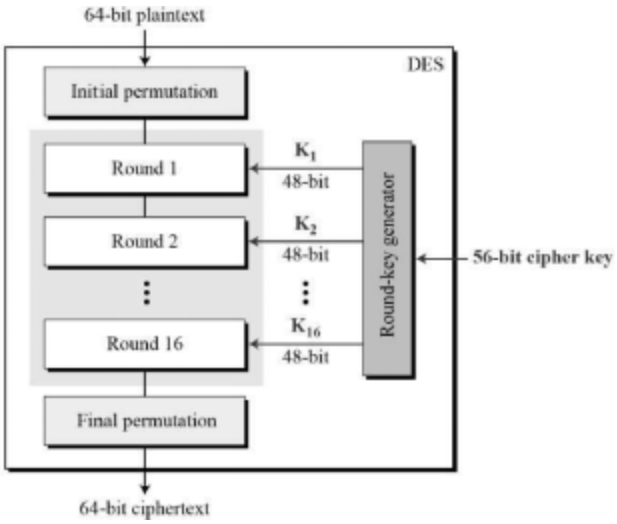
6. **Threat Modeling**: Document, locate, address, and validate are the four steps to threat modeling. To securely code, you need to examine your software for areas susceptible to increased threats of attack.

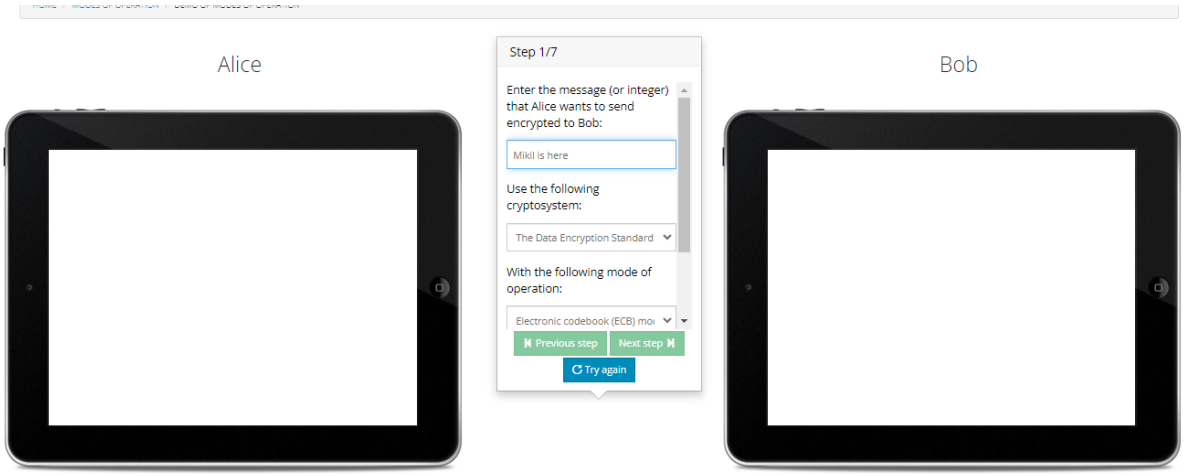
7. **Cryptographic Practices**: Encrypting data with modern cryptographic algorithms and following secure key management best practices increases the security of your code in the event of a breach.

8. **Input Validation and Output Encoding**: These secure coding standards are self-explanatory in that you need to identify all data inputs and sources and validate those classified as untrusted. You should utilize a standard routine for output encoding and input validation.

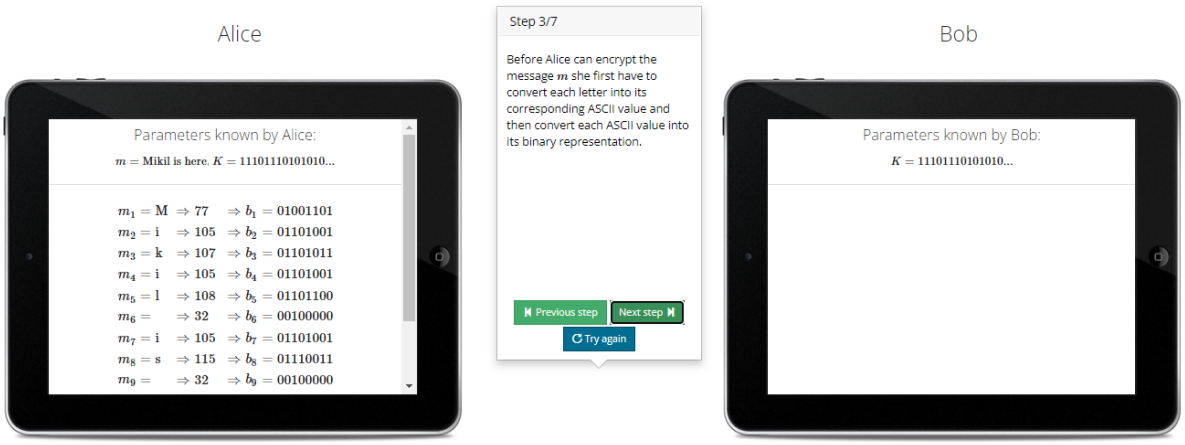
Symmetric Cryptography Demonstration

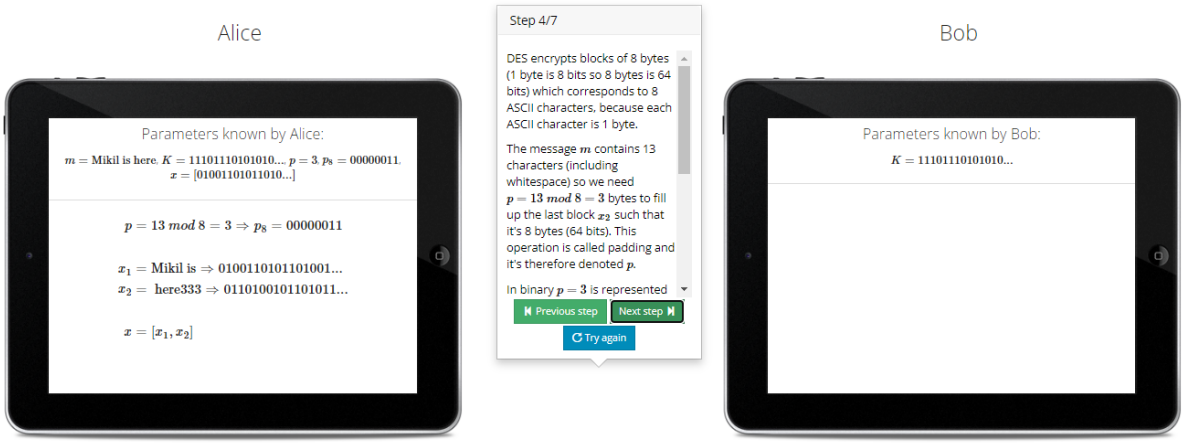
DES is a block cipher and encrypts data in blocks of size of 64 bits each, which means 64 bits of plain text go as the input to DES, which produces 64 bits of ciphertext.

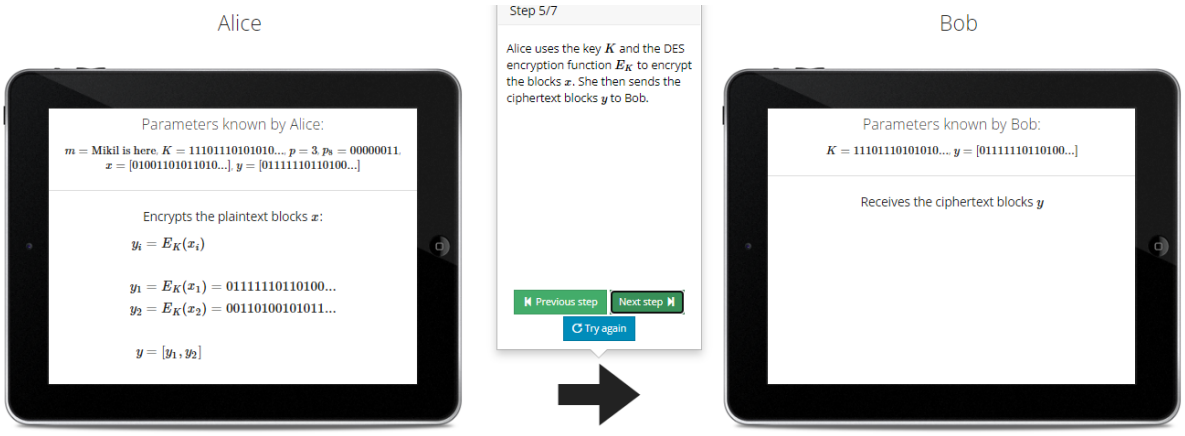

64-bit plaintext
DES
Initial permutation
Round 1
Round 2
Round 16
Final permutation
K1
48-bit
K2
48-bit
K16
48-bit
Round-key generator
56-bit cipher key
64-bit ciphertext

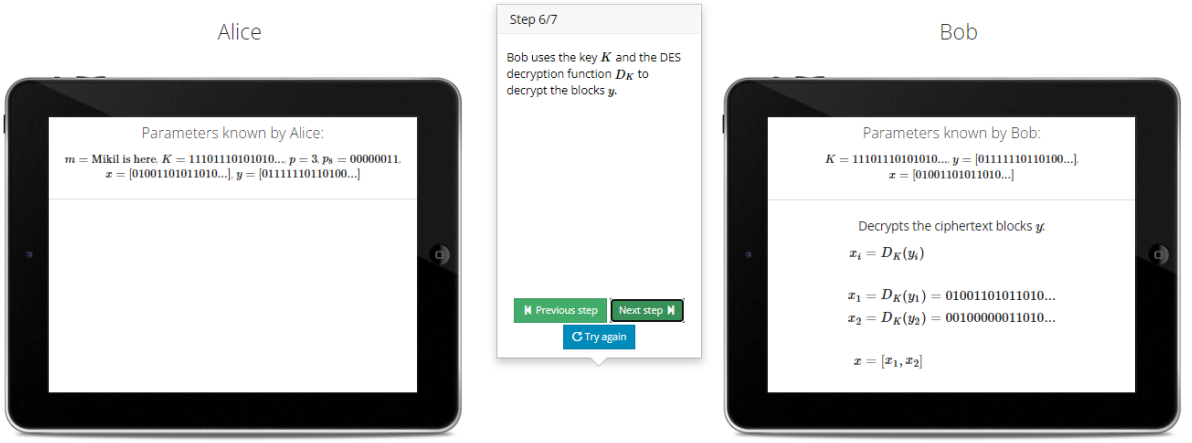

Alice
Step 1/7
Enter the message (or integer) that Alice wants to send encrypted to Bob:
Mikil is here
Use the following cryptosystem:
The Data Encryption Standard
With the following mode of operation:
Electronic codebook (ECB) mo
Previous step
Next step
Try again
Bob

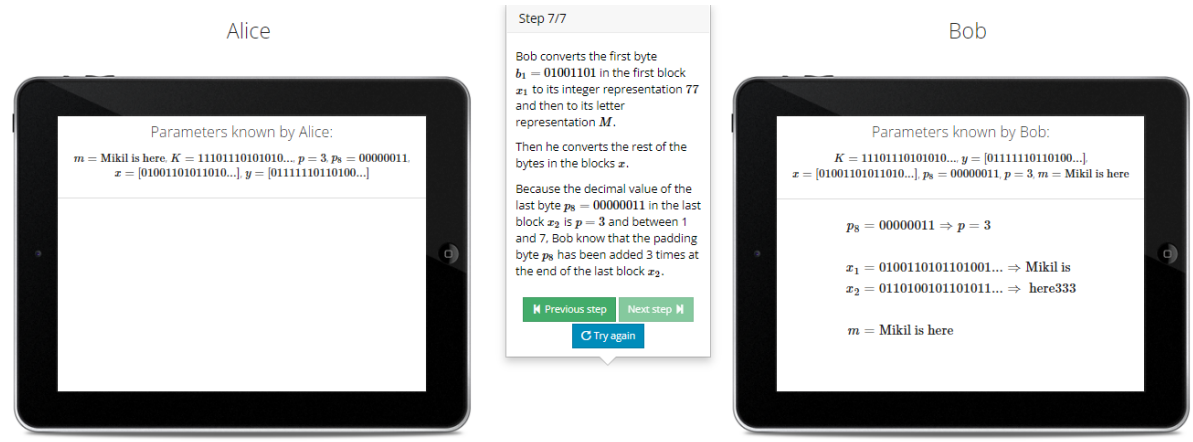

Alice
Parameters known by Alice:
m = Mikil is here, K = 11101110101010...
Chooses the key K
K = 11101110101010...
Step 2/7
Alice chooses the 56-bit key K for the message m.
She then sends the key K through a secure channel to Bob.
Previous step
Next step
Try again
Bob
Parameters known by Bob:
K = 11101110101010...
Receives the key K


Alice
Parameters known by Alice:
m = Mikil is here, K = 11101110101010...
m_1 = M ⇒ 77 ⇒ b_1 = 01001101
m_2 = i ⇒ 105 ⇒ b_2 = 01101001
m_3 = k ⇒ 107 ⇒ b_3 = 01101011
m_4 = i ⇒ 105 ⇒ b_4 = 01101001
m_5 = l ⇒ 108 ⇒ b_5 = 01101100
m_6 = ⇒ 32 ⇒ b_6 = 00100000
m_7 = i ⇒ 105 ⇒ b_7 = 01101001
m_8 = s ⇒ 115 ⇒ b_8 = 01110011
m_9 = ⇒ 32 ⇒ b_9 = 00100000
Step 3/7
Before Alice can encrypt the message m she first have to convert each letter into its corresponding ASCII value and then convert each ASCII value into its binary representation.
Previous step
Next step
Try again
Bob
Parameters known by Bob:
K = 11101110101010...


Alice
Parameters known by Alice:
m = Mikil is here, K = 11101110101010..., p = 3, p_8 = 00000011, x = [01001101011010...]
$p = 13 \ mod \ 8 = 3 \Rightarrow p_8$ = 00000011
x_1 = Mikil is ⇒ 0100110101101001...
x_2 = here333 ⇒ 0110100101101011...
$x = [x_1, x_2]$
Step 4/7
DES encrypts blocks of 8 bytes (1 byte is 8 bits so 8 bytes is 64 bits) which corresponds to 8 ASCII characters, because each ASCII character is 1 byte.
The message m contains 13 characters (including whitespace) so we need $p = 13 \ mod \ 8 = 3$ bytes to fill up the last block x_2 such that it's 8 bytes (64 bits). This operation is called padding and it's therefore denoted p.
In binary $p = 3$ is represented
Previous step
Next step
Try again
Bob
Parameters known by Bob:
K = 11101110101010...


Alice
Parameters known by Alice:
m = Mikil is here, K = 11101110101010..., p = 3, p_8 = 00000011, x = [01001101011010...], y = [01111110110100...]
Encrypts the plaintext blocks x:
$y_i = E_K(x_i)$
$y_1 = E_K(x_1)$ = 01111110110100...
$y_2 = E_K(x_2)$ = 00110100101011...
$y = [y_1, y_2]$
Step 5/7
Alice uses the key K and the DES encryption function E_K to encrypt the blocks x. She then sends the ciphertext blocks y to Bob.
Previous step
Next step
Try again
Bob
Parameters known by Bob:
K = 11101110101010..., y = [01111110110100...]
Receives the ciphertext blocks y





Asymmetric Cryptography Demonstration

RSA algorithm is an asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. Public Key and Private Key.

Step 1: Generate the RSA modulus

The initial procedure begins with selection of two prime numbers namely p and q, and then calculating their product N, as shown: *N=p*q*

Here, let N be the specified large number.

Step 2: Derived Number (e)

Consider number e as a derived number which should be greater than 1 and less than (p-1) and (q-1). The primary condition will be that there should be no common factor of (p-1) and (q-1) except 1

Step 3: Public key

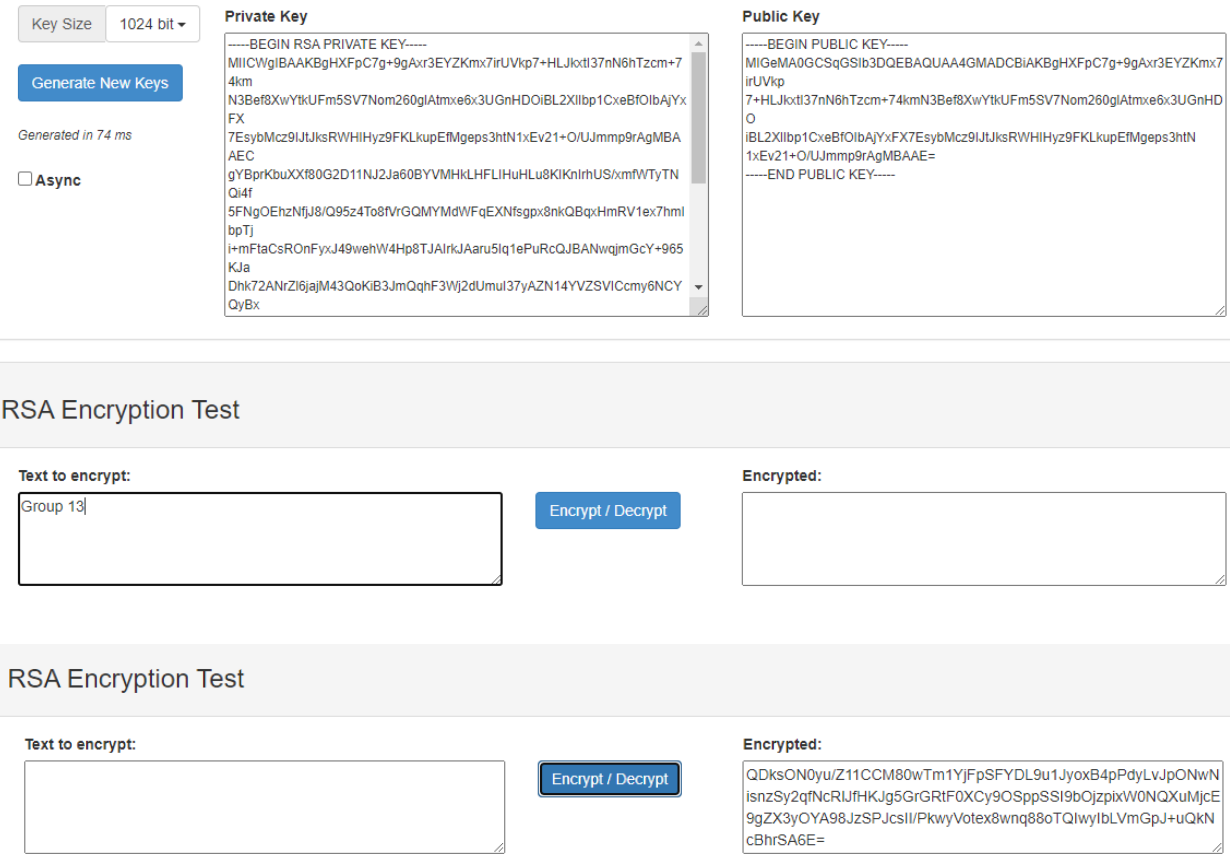
The specified pair of numbers n and e forms the RSA public key and it is made public.

Step 4: Private Key

Private Key d is calculated from the numbers p, q and e. The mathematical relationship between the numbers is as follows: *ed = 1 mod (p-1) (q-1)*

The above formula is the basic formula for Extended Euclidean Algorithm, which takes p and q as the input parameters.

Decryption Formula: The decryption process is very straightforward and includes analytics for calculation in a systematic approach. Considering receiver C has the private key d, the result modulus will be calculated as *Plaintext = Cd mod n*



Conclusion:

Thus we have studied how to perform different types of Symmetric and Asymmetric cryptography.