# Applied Cryptography

**Week #3 Extra**

Bernardo Portela and Rogério Reis

2024/2025

**Important**

- Your answers must **always** be accompanied by a justification. Presenting the final result (e.g. the result of a calculation) without the rationale that laid to said result will result in a grade of 0.
- Submit your answers via e-mail to *bernardo.portela@fc.up.pt*, with adequate identification of the group and its members.

## Q1: Weak Security

Unpredictability of key generation is a central requirement to the security of an encryption scheme. If the key can be efficiently guessed, then no encryption scheme can ever be shown to be IND-CPA secure, as any adversary can simply enumerate the possible keys and test for decryptions.

The code `ciphersuite_aesnotrand.py` is encrypting a block message using a very weak key. Check it out to understand what it is doing wrong.

**Question - P1:** Program `q1.py` produces `weak_ciphertexts`. Suppose you know that the encrypted message was "Attack at Dawn!!' ". Extend that program to read the file and guess the key used for that encryption

**Question - P2:** Increase the size of the `offset` in the ciphersuite. How large must it be for your machine to be unable to test it in 3 hours?

## Q2: Fixed Initialization Vectors

Figure 1 depicts the Chaining Block Mode (CBC). One key characteristic of AES-CBC (AES as block encryption, used in combination of CBC) is that it requires for initialization vectors to be **unique** and **unpredictable**

Recall the IND-CPA experiment:

- Challenger generates a secret key $k$ and a random bit $b$
- Attacker can send $m$ and receive $\mathsf{Enc}(k, m)$ – has access to an encryption oracle
- Attacker provides $(m_0, m_1)$ such that $|m_0| = |m_1|$ and receives $\mathsf{Enc}(k, m_b)$
- Attacker guesses $b'$
- Attacker is victorious if $b = b'$.

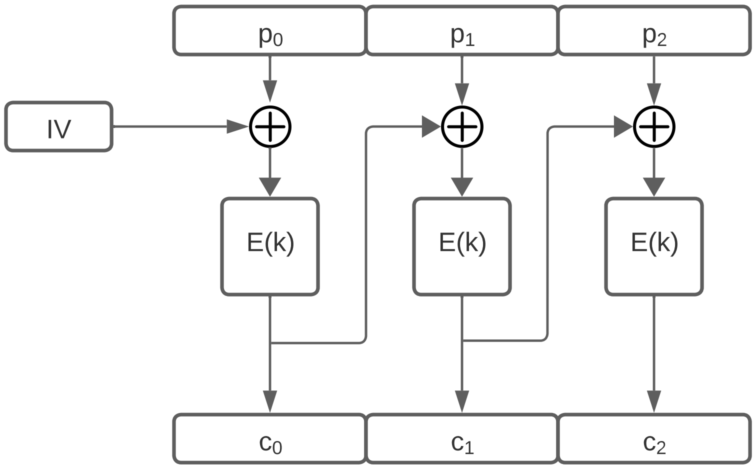


Figure 1: AES-CBC encryption mode

Scheme is broken if this occurs with non-negligible probability over $\frac{1}{2}$

**Question:** Suppose our encryption scheme is AES-CBC using a **fixed IV**. Construct an attack against the IND-CPA security experiment of this scheme, i.e. write an algorithm for our adversary to beat the IND-CPA security experiment, namely:

- What are the queries performed to the encryption oracle
- What are the messages produced as $m_0, m_1$
- How $b$ is decided

## Q3: Predictable Initialization Vectors

Nonce-based encryption schemes are encryption schemes that take the *nonce* as a parameter. $\mathsf{Enc}(k, m, n)$ takes key $k$, message $m$ and nonce $n$. These are secure, as long as no *nonce* is ever used twice. E.g. AES-CTR is a nonce-based encryption scheme.

Consider the following encryption scheme:

- Use the block encryption function (with the same key) on the nonce to generate an IV $\leftarrow E(k, n)$
- Compute the encryption of the message using AES-CBC with that IV

Observe that this prevents trivial attacks, such as setting the IV to 0 – as it is encrypted – and also disallows fixing the IV – as the same nonce cannot be reused. However, the IV is **predictable**, and that can lead to an attack.

**Question - P1:** Construct an attack against the nonce-based IND-CPA security experiment of this scheme[1]

*Hint:* Consider encrypting $0^l$ with nonce $0^l$. How can I request a correlated encryption that can help me break the indistinguishability of the cipher?

**Question - P2:** Write a program that prints the messages/ciphertexts used in this attack, and that shows this IND-breaking correlation.

[1]Nonce-based IND-CPA is exactly the same as IND-CPA, but repeated nonces are disallowed.

## Q4: Padding Attacks

Encryption schemes such as AES-CBC can encrypt messages of varying size, by dividing the input message into chunks of size $b$, where $b$ is the block size. However, it is common for messages to not be multiples of $b$, and for these cases one can use *padding*.

Let $k$ denote the next multiple of $b$ for the message $m$. PKCS#7 padding entails filling the last $k - |M|$ bytes with value $k - |M|$, e.g.

- `0x01` means 1 byte of padding added with this value
- `0x03` means 3 bytes of padding added with this value

**Question - P1:** Consider a message that is already of size multiple of $b$. Why is it necessary to add padding?

**Question - P2:** Consider an AES-CBC encryption scheme that, upon decryption, produces an error whenever a padding error occurs, i.e. if the decrypted message does not follow PKCS#7 padding.

How can an adversary that is given a ciphertext use a decryption oracle to extract information about the original message?

*Hint:* Consider how AES-CBC decrypts messages. How can we provoke alterations on the last block, where padding must be observed?