# TP1 – Cryptopals avec Python


## M1-SI – Module : Sécurité avec Python

**Date :** 16 Juin 2025

**Enseignant :** *Arij AZZABI*

# Objectifs pédagogiques

- Comprendre les primitives cryptographiques (XOR, AES, CBC...).
- Implémenter des fonctions de chiffrement / déchiffrement en Python.
- Identifier des schémas de chiffrement faibles comme ECB.
- Appréhender des attaques comme le padding oracle ou la détection de mode.

# Contexte

Vous êtes analyste sécurité et souhaitez explorer les fondements de la cryptographie appliquée. Les challenges Cryptopals permettent d'implémenter des fonctions cryptographiques en Python tout en comprenant leurs forces et faiblesses.

# 1 Environnement technique

- **Python 3.8+**
- Éditeur : VSCode, PyCharm ou terminal
- Modules autorisés : `binascii`, `base64`, `cryptography` (si besoin)
- Fichier : `tp1_cryptopals.py`

# 2 Travail à réaliser

## Set 1 – Niveau débutant à intermédiaire

- **Challenge 1 :** Convertir hex → base64
- **Challenge 2 :** XOR entre deux buffers
- **Challenge 3 :** Déchiffrer XOR simple
- **Challenge 4 :** Détecter la ligne chiffrée en XOR simple
- **Challenge 5 :** Repeating-key XOR
- **Challenge 6 :** Briser repeating-key XOR (attaque)
- **Challenge 7 :** AES-ECB
- **Challenge 8 :** Détecter ECB automatiquement

**NB :** Les challenges doivent être codés **manuellement** en Python, sans copier de solution en ligne. L'utilisation d'une bibliothèque est autorisée pour AES.

### Set 2 – Concepts avancés (bonus)

- Compréhension du padding PKCS#7
- ECB cut-and-paste attack
- CBC bit flipping attack
- Oracle padding (démo ou annexe possible)

## 3 Livrables attendus

- Script Python bien structuré : `tp1_cryptopals.py`
- Un fichier `README.md` expliquant vos choix d'implémentation
- Un court rapport PDF répondant aux questions :
  - Quels types d'attaques avez-vous pu simuler ?
  - Quel est l'intérêt de détecter l'usage de ECB ?
  - Quelles erreurs de sécurité sont illustrées ?