Pràctica 5: Signatura d'anell

Funcions útils per realitzar la pràctica:

- `UAB_E(skey, plain_bytes)`
  Realitza el xifrat simètric de `plain_bytes` utilitzant AES amb la clau `skey`.
- `UAB_E_inv(skey, enc_bytes)`
  Realitza el desxifrat de `enc_bytes` amb AES.
- `UAB_h(m)`
  Hash SHA-256 de `m`.
- `UAB_xor(a, b)`
- `UAB_generate_RSA_public_key(nBits)`
  Retorna una tupla `(n, e)` corresponent a una clau pública RSA. `nBits` és la mida del mòdul `n`.
- `UAB_generate_RSA_key_pair(nBits)`
  Retorna una tupla `(pk, sk)` corresponent a un parell clau pública - clau privada RSA.
- `os.urandom(size)`
  Retorna una cadena aleatòria de bytes de mida `size`.

En aquesta pràctica, ens posarem en la pell d'un treballador d'una central nuclear que vol difondre un missatge alertant d'una situació de perill sense revelar la seva identitat, però demostrant als receptors que el missatge ha estat enviat per un operador de la central. Per fer això, utilitzarem les signatures d'anell. D'aquesta manera, la premsa sabrà que la denúncia prové d'un empleat amb accés a la informació de la central, però no es podrà saber quin dels empleats en concret ha generat la signatura.

1. (**2 p**) Implementeu les següents funcions de xifrat RSA.

   (a) UAB_f(PK,x): Xifra l'objecte bytes `x` utilitzant RSA. `PK` és una tupla `(n, e)` corresponent a la clau pública. Tot i calcular el valor xifrat utilitzant enters, aquesta funció retornarà el resultat com un objecte bytes. Com en les pràctiques anteriors, utilitzeu les funcions `from_bytes` i `to_bytes` per passar de `bytes` a enters i d'enters a `bytes`.

   (b) UAB_f_inv(SK,y): Desxifra l'objecte bytes `y` utilitzant RSA. `SK` és una tupla `(n, d)` corresponent a la clau privada. Tot i calcular el valor en clar utilitzant enters, aquesta funció retornarà el resultat com un objecte bytes.

   A l'hora d'implementar aquestes funcions, haureu d'aplicar el mecanisme definit per aquest protocol de signatura d'anell que permet xifrar i desxifrar valors que són més grans que els mòduls `n` de les claus RSA. Per tant, les funcions `UAB_f` i `UAB_f_inv` aplicaran l'algorisme de xifrat o desxifrat de la manera següent:

$$g(m) = \begin{cases} q \cdot n + f(r) & \text{si } (q+1) \cdot n \leq 2^b \\ m & \text{altrament} \end{cases}$$

on f és la funció de xifrar o desxifrar, m és el valor a tractar en les respectives funcions (paràmetres `x` i `y`), i q, r són el quocient i el residu de m/n.

Observeu que la mida en bits (b) ja ve definida a la pràctica mitjançant la variable `v_bits=1024`. Aquesta variable està declarada al principi de la plantilla i heu de tenir en compte que:

- pot tenir un valor ≥ 1024, i
- ha de ser múltiple de 128, per la forma com teniu implementat el xifrat simètric AES.

2. (**0,5 p**) Definiu totes les variables que necessitareu per poder realitzar la signatura d'anell.

   - `LENGTH`: mida de l'anell, és a dir, nombre d'operadors que poden haver generat la signatura.
   - `position`: posició del signant dins de la llista `xs`, `ys` i `publicKeysGroup`.
   - `xs`: llista amb un valor aleatori x per a cada membre del grup, segons indica aquest protocol. A la posició del signant, contindrà el valor `None`.
   - `pkSigner, skSigner`: parella de claus pública-privada RSA del signant. Utilitzeu la funció `UAB_generate_RSA_key_pair` amb una mida de 1024 bits.
   - `publicKeysGroup`: llista amb les claus públiques dels membres del grup. Utilitzeu la funció `UAB_generate_RSA_public_key` amb una mida de 1024 bits i introduïu la clau pública del signant a la posició corresponent.
   - `ys` llista amb el resultat d'aplicar la funció `UAB_f` als valors de `xs`, també amb un `None` a la posició del signant.

3. (**3,5 p**) Implementeu la funció `UAB_solve_C(k,v,ys)` per obtenir el valor de y per al signant. Aquest valor serà el que permetrà que la sortida de la funció de combinació sigui igual al valor d'inicialització v. És a dir:

$$C_{k,v}(y_1, y_2, \cdots, y_r) = v$$

   Per fer-ho, caldrà partir del valor inicial v i seguir l'anell en sentit horari i en sentit antihorari per arribar a obtenir els valors de la XOR que permeten calcular la y del signant.

4. (**1,5 p**) Implementeu la funció `UAB_sign_ring(m, v)` per construir la tupla σ que conté el resultat de la signatura. Aquesta tupla inclourà:

   - llista de claus públiques dels membres del grup
   - vector d'inicialització
   - llista de valors x de tots els membres

   Per completar la llista `xs`, aquesta funció utilitzarà la funció hash `UAB_h` per calcular la clau del xifrat AES. A més, utilitzareu la funció `UAB_solve_C` per trobar el valor y del signant i, amb aquest, trobar el valor x corresponent que s'inclourà a la llista `xs` final.

5. (**2,5 p**) Implementeu la funció `UAB_verify_ring_signature(m, sigma)` que verifica la signatura d'anell d'un missatge m. Haureu de calcular la clau k, la llista de valors y i, finalment, el resultat de la funció de combinació per comprovar que el valor z final és igual a v. En aquest cas, la funció retornarà `True`.